CA SOLVE:Access™ Session Management

Administration Guide

r5

CA Technologies Product References

This document references the following CA Technologies products:

- CA NetMaster® Network Management for TCP/IP (CA NetMaster NM for TCP/IP)
- CA NetSpy™ Network Performance (CA NetSpy)
- CA SOLVE:Access™ Session Management (CA SOLVE:Access)

Contact CA Technologies

Contact Technical Support

For your convenience, CA Technologies provides one site where you can access the information you need for your Home Office, Small Business, and Enterprise CA Technologies products. At http://ca.com/support, you can access the following:

- Online and telephone contact information for technical assistance and customer services
- Information about user communities and forums
- Product and documentation downloads
- CA Support policies and guidelines
- Other helpful resources appropriate for your product

Provide Feedback

If you have comments or questions about CA Technologies product documentation, you can send a message to techpubs@ca.com.

If you would like to provide feedback about CA Technologies product documentation, complete our short customer survey, which is available on the CA Support website at http://ca.com/docs.

Contents

Chapter 1: Introduction

This section contains the following topics:

Intended Audience

This guide is intended for technical personnel responsible for the planning and maintenance of the CA SOLVE:Access components (EASINET and MAI) on your system. You must have experience with CA SOLVE:Access and Operator Console Services (OCS).

Note: To find out what you can do with EASINET and MAI or learn how to use these features, see the *User Guide*.

Notational Conventions

This section explains the conventions used when referring to various types of commands and when indicating field attributes.

Commands

Commands such as SYSPARM and SHUTDOWN are shown in upper case.

User Entries

Information to be entered onto panels displays in bold text.

Cross-References

Cross-reference links to other sections of the book are displayed as underlined teal text.

Components

CA SOLVE:Access consists of the following components:

EASINET Enhanced Access and Security Interface

Provides a user-friendly and secure logon screen as well as the facility to control idle terminals, broadcast messages to network users and simplify access to applications.

MAI Multiple Application Interface

Provides easy access to the application you require and the facility to operate multiple applications concurrently.

EASINET—Enhanced Access and Security Interface

EASINET provides an easy logon procedure and guidance to the applications which are available to you.

Network terminals provide logical connections (or sessions) to a variety of different application programs. Therefore a terminal must be in an idle state when not connected to an application program, and not permanently associated with a single application. The only function supported while in the idle state is the capacity to log on to a selected application. By default, terminals in the idle state are under VTAM control.

The terminal you see usually displays some form of logo defined to VTAM by table generators (from USS tables, interpret tables). VTAM facilities allow limited customization of logo and messages, but do not allow device access for issuing broadcast messages or any function, other than entering logon requests. This severely limits the scope of communication with network users and makes it impossible to impose security verification of network users at the boundary of the network.

EASINET Functions

EASINET is a system-level network control language (NCL) procedure that performs the following functions:

- Provides a default mechanism which brings idle display terminals in an installation under control.
- Allows broadcast messages to be sent throughout your installation to warn of operational problems or other events.
- Greatly simplifies the definition and control of logon procedures.
- Eliminates the need for complex and inflexible USS tables and interpret tables.

The EASINET facility operates only while CA SOLVE:Access is running in the system. Therefore, if CA SOLVE:Access is inactive for any reason, EASINET terminals will be under the normal control of VTAM USS services. For this reason, it can be helpful for you to have a minimum configuration USS table to support terminals on the network.

MAI—Multiple Application Interface

The Multiple Application Interface (MAI) is a feature of CA SOLVE:Access that allows a user at a single terminal to operate more than one session at once. It also has other features, for example, MAI can be used to record anything that displays on a terminal's screen.

MAI operates in either of the following modes:

- Full Screen mode: MAI-FS
- Operator Control (line-by-line) mode: MAI-OC

The principal difference between the two modes is their display on the screen; MAI-FS uses the entire screen and MAI-OC uses just one line at a time.

MAI Full Screen Mode (MAI-FS)

MAI-FS allows you to use one terminal to establish sessions with any number of VTAM application systems (for example, IMS, TSO, or CICS) in addition to performing NetMaster® functions.

MAI-FS also allows you to terminate sessions, jump to specific sessions and modify your MAI-FS operating environment. MAI-FS also allows:

- Automated establishment and control of sessions without the need for intervention from the terminal operator
- Non-terminal-owning regions to establish and control MAI-FS sessions

Note: MAI-FS sessions use the Logical Unit Type-2 (LU.2) protocol, appearing to the application as a device equivalent to an IBM 3278/9 display terminal, or the Logical Unit Type-0 (LU.0) protocol, appearing to the application as a device equivalent to an IBM 3277 display terminal. MAI-FS usually emulates the device from which it is invoked, and supports all models and sizes of terminal, including those terminals that support graphics and color.

Session Automation With a Script

You can establish or operate an MAI-FS session automatically using a script. A script is a set of instructions written in NCL. For instance, a script executed at the start of the session could automatically enter a user ID and password, and then enter the session's first command. In this manner, logons to all applications can be automated, providing a single sign-on capability. Scripts can also be used during session activity to perform functions and to terminate sessions.

MAI Operator Control Mode (MAI-OC)

MAI Operator Control mode (MAI-OC) is available from an Operator Console Services (OCS) window or from an NCL procedure. As with MAI-FS, you can start multiple sessions with VTAM applications - the difference is that an MAI-OC session operates using Logical Unit Type-1 (LU.1) protocols, and appears to the application as a line-by-line device such as an IBM 3767 terminal.

In the case of MAI-OC sessions operated from an OCS window, output received across the sessions displays line-by-line on the user's OCS window, together with any other SOLVE output. MAI-OC can be used by a SOLVE operator to provide centralized operation of both the network (through normal OCS functions) and of major systems such as CICS, IMS or JES, where the MAI-OC sessions act as the master consoles of the other application systems.

Session Replay Facility (SRF)

The Session Replay Facility (SRF) provides the ability to capture the screen output of any number of session dialogs to a database, and then, to replay that output from another terminal.

SRF is an invaluable help-desk aid. Sessions can be replayed exactly as the user saw them, so that user errors or application errors are readily identified, so that help can be given to the user. Session dialogs of any terminal connected to CA SOLVE:Access can be captured in this way, whether the terminal is being used for an MAI session or any other NetMaster function.

Screen Image Services Facility (MAI-SIS)

The MAI Screen Image Services (MAI-SIS) feature provides functions to transfer screen data between sessions or within one session. These functions are performed with special MAI commands or from an MAI Screen Image Services menu panel.

MAI : Primary Menu

A feature of CA SOLVE:Access is NCL Panel Services. The MAI : Primary Menu is also NCL-based. This allows you to customize these menus or write your own menu. In this way, the menu's format can be designed to integrate with the rest of your system.

Chapter 2: Starting and Stopping a Region

This section contains the following topics:

Start a Region

To start a region, you need to run it as a job or a started task. A started task should have been set up during the installation process.

To start a region, issue the following command:

```
S rname
```

Users log on to a region by using the user IDs and passwords specified in their UAMS (or external security package) records.

Stop a Region

If you have the necessary authority, you can shut down the region by issuing the SHUTDOWN or FSTOP command.

SHUTDOWN Command

The SHUTDOWN command stops the region when the last user logs off. When you issue the SHUTDOWN command, a broadcast is issued to all users. No further logons are accepted until the region is restarted, or the SHUTDOWN CANCEL command is issued.

You can issue the SHUTDOWN command from OCS or Command Entry. Alternatively, you can issue it as a z/OS MODIFY command.

Note: For more information about the SHUTDOWN command, see the online help.

FSTOP Command

The FSTOP command immediately disconnects user sessions and shuts down the region.

Restrict the use of the FSTOP command.

You can issue the FSTOP command from OCS or Command Entry. Alternatively, you can issue it as a z/OS MODIFY command.

Important! If you are running another product in the same region, it also stops if the FSTOP command is issued.

Note: For more information about the FSTOP command, see the online help.

Preserve Data When Region Stops and Restarts

You may want to preserve some data when a region stops so that this data is available when the region restarts. You can use global variables to preserve data. You can save global variables that the region reloads when it restarts. Saved global variables are known as persistent global variables.

To preserve data, create global variables with data you want to preserve and save them, for example:

- Use the Persistent Variables Administration option (access shortcut is /PVARS).
- Call the $CAGLBL procedure using the SAVE option.

  Note: For information about the $CAGLBL procedure, see the *Network Control Language Reference Guide*.

Create Persistent Global Variables Using the User Interface

You can create persistent global variables from the Persistent Variables List panel. The panel also lets you maintain those variables, for example, update, purge, or reload them.

To create a persistent global variable using the user interface

1. Enter the **/PVARS** panel shortcut.

   The Persistent Variables List panel appears.

2. Press F4 (Add).

   The Persistent Variable - Add panel appears.

3. Specify the name of the variable (without its global prefix) and its value. Press F3 (File).

The variable is saved so that it can be loaded the next time the region starts up.

Prevent the Reloading of Preserved Data

If problems occur during region startup because of invalid data being loaded, you can disable the reloading of the preserved data.

To prevent the reloading of preserved data, enter the following command when you start the region:

```
S rname, PARM='XOPT=NOPVLOAD'
```

The region starts without reloading the preserved data.

Chapter 3: Configuring a Region

This section contains the following topics:

Use JCL Parameters to Configure a Region

JCL parameters enable you to configure a region. You use JCL parameters to set information such as the names of your INIT and READY procedures, and the types of security exit to use in your region.

This information is supplied by the PPREF statements in the RUNSYSIN member.

You can also pass this information in the START command using the JCL PARM field. If you specify multiple parameters, separate each with a comma.

Note: For more information, see the *Reference Guide*.

Display and Change JCL Parameter Settings

You can display the current settings of all the JCL parameters with the SHOW PARMS command from OCS or Command Entry. To change any of these parameters, specify their new values in the RUNSYSIN member and then restart the region.

Note: For more information about JCL parameters, see the *Reference Guide*.

Identify the Region to Users

If you have multiple regions or communicate with other regions, you can set the domain ID and put titles on the panels.

Identify Domains and Panels

The NMDID JCL parameter identifies the domain ID for each region. If you have multiple regions, ensure that you have a different domain ID for each one.

Note: For more information about the NMDID parameter, see the *Reference Guide*.

You can use the SYSTEMID (System Identifications) parameter group in Customizer to help identify your regions. This parameter group specifies a system identifier that is used when you link to other regions. Ensure that each of your regions has a different system identifier.

This parameter group also specifies the titles to display on the logon panel and the OCS console panel. These titles help users to identify the region that they have logged on to.

Note: The system ID parameter takes effect when the region is initialized.

Customize a Region Using Customizer

Customizer lets you review and update parameter groups.

You use Customizer to initialize and customize your region. Customizer is an initialization facility that lets you implement a region rapidly and easily. Also, Customizer enables you to customize parameters easily at a later stage.

When you first install a product, you need to set various parameters to get the product up and running. Customizer helps you set up these parameters. An initial dialog is supplied for the first time user, to walk you through the customization process. You are prompted to supply required parameter values and given the opportunity to supply optional parameter values.

To access the parameter groups, enter **/PARMS**.

What Are Parameter Groups?

System parameters are grouped by category (such as Security) in logical parameter groups, to simplify the process of initializing and customizing a region.

Groups of individual parameters translate into one or more of the following:

- SYSPARMS that determine how your region functions
- Global variables that are used by various NCL applications to control their functions
- Local parameters that define how to implement actions associated with parameter groups

Update System Parameters

Most customization of your region is performed by using Customizer.

You can also use the SYSPARMS command to customize your region. Each operand of the SYSPARMS command lets you specify options to change and customize the way your region works. For ease of maintenance, you can use the Display/Update SYSPARMS panel, which is accessible by using the /SYSPARM panel shortcut.

Notes:

- SYSPARMS set by Customizer parameter groups can only be updated using Customizer.
- For SYSPARMS without a corresponding parameter group, set the SYSPARMS in the INIT and READY procedures so that they are applied when the region starts, and then update them dynamically using the SYSPARMS command.
- For more information about SYSPARMS operands, see the *Reference Guide*.

Use the SYSPARMS Command

To change a SYSPARMS operand with the SYSPARMS command, enter the command at the OCS command line.

This command has the following format:

```
SYSPARMS operand=value operand=value ...
```

Example: Use the SYSPARMS Command

To display the time at the beginning of the OCS title line, enter the following command:

```
SYSPARMS OCSTIME=YES
```

Initialization Operands

There are some SYSPARMS command operands that cannot be changed while the region is operational. These operands must be included in your INIT procedure so that they are executed during initialization.

Note: For a complete list of SYSPARMS commands, see the *Reference Guide*.

If you specify new values for these initialization operands, the new values do not take effect until the region is initialized. All other SYSPARMS can be changed during region operation by authorized users.

Chapter 4: Administering This Product

This section contains the following topics:

Overview

To enable the CA SOLVE:Access facilities, you perform various administrative tasks to implement and customize the facilities to suit your installation. Administrative tasks are performed using one of the following components:

- Customizer
- SNA Access Services Administration

Customizer

Many of the operational aspects of CA SOLVE:Access may be customized using Customizer. These operational considerations are grouped together to form parameter groups that are defined to Customizer using unique names, and are further grouped by category.

Customizer is used to perform implementation tasks, including:

- Implementing Applications
- Implementing and Administering EASINET
- Implementing and Administering MAI
- Implementing Terminal Security
- Implementing Support for Generic Resources

Customizer parameter groups are executed when CA SOLVE:Access is initialized to customize the product to your requirements when the region is started. You can also use Customizer after initialization to:

- Change parameters for immediate effect. To do this, the parameter group is actioned.
- Change parameters for the next time that the region is started. To do this, the parameter is updated.
- Action and update parameter groups for immediate effect and on region restart.

To use Customizer

1. Enter **/PARMS** on any panel.
2. Enter **U** next to the required parameter group.
3. Change the fields as required. Press F1 (Help) for help.

   Some parameter groups have multiple pages. Use the F10 and F11 keys to move between the pages.
4. To action the parameter group, press F6 (Action).
5. To update the parameter group, press F3 (File).

SNA Access Services Administration

The SNA Access Services Administration Menu is used to perform other administration tasks including:

- Using the MAI Stored Definition Maintenance (MSDM) facility to update individual user's MAI session lists, or to make global changes
- Updating MAI session defaults
- Generating logon scripts
- Maintaining EASIUSER definitions
- Implementing support for IBM NetView™ Synergy Interface (NSI)

To use the SNA Access Services Administration Menu

1. Enter **/ACADMIN** on any panel.

   The SNA Access Services : Administration Menu displays.
2. Select the required option. Press F1 (Help) for help.

Implement Applications

To enable access to applications, for example, TSO or CICS, you define logon paths that identify the target applications and operational parameters. These logon paths are defined in the DEFLOGON table, which contains a set of DEFLOGON entries created by the execution of DEFLOGON commands.

These commands are defined in your CA SOLVE:Access initialization procedure. The name of this procedure is specified in the ACINIT parameter group of Customizer with a default name of $ACINIT.

To specify another name

1. Enter **/PARMS** on any panel.
2. Enter **U** next to the $AC ACINIT parameter group in the Names category.
3. Enter the name of the procedure in the User Initialization Procedure field. Press F1 (Help) for online help.
4. To action the parameter group, press F6 (Action).
5. To update the parameter group, press F3 (File).

More information:

Defining and Administering Applications (see page 81)

How You Implement and Administer EASINET

To implement and administer EASINET, perform the following tasks:

- Specify the EASINET procedure name.
- Define logon user data requirements.
- If you use EASIUSER, define the EASIUSER database and add EASIUSER definitions to this data set.

Specify the EASINET Procedure Name

EASINET is implemented using an NCL procedure. This implementation enables complete flexibility when designing screen panels and provides logic capability, access to files, and access to security system information. EASINET provides a simple way to customize the presentation of the network to users.

CA SOLVE:Access is distributed with the following working samples of an EASINET system:

- The $EASINET NCL procedure.
- A suite of NCL procedures and panels that make up the EASIUSER application.
- Alternatively, you can develop your own EASINET system.

The name of your EASINET procedure must be defined to CA SOLVE:Access. By default, CA SOLVE:Access uses the $EASINET procedure.

To specify an alternative procedure name

1. Enter **/PARMS** on any panel.
2. Enter **U** next to the $AC ACINIT parameter group in the Names category.
3. Enter the required value in the EASINET Procedure field. Press F1 (Help) for online help.
4. To action the parameter group, press F6 (Action).
5. To update the parameter group, press F3 (File).

More information:

Implementing EASINET (see page 43)

Define Logon User Data Requirements

The LOGONUSRDATA parameter groups allow you to control whether user data is accepted when logging on to this region.

To define logon user data requirements

1. Enter **/PARMS** on any panel.
2. Enter **U** next to the $NM LOGONUSRDATA parameter group in the Security category.
3. Enter the required value in the Accept User Data at Logon? Field. Press F1 (Help) for online help.
4. To action the parameter group, press F6 (Action).
5. To update the parameter group, press F3 (File).

Implement the EASIUSER System

EASIUSER is an extension to the EASINET function that allows you to customize a list of applications for an individual or groups of users. To use the EASIUSER system:

- Define the EASIUSER definition database (EASIUDB).
- Add application definitions, group definitions, and user definitions to the EASIUDB data set.

To allocate the EASIUDB

1. Enter **/PARMS** on any panel.
2. Enter **U** next to the $AC EASIUDB parameter group in the Files category.
3. Enter the data set name in the Dataset Name field. If you do not specify the data set name, the file is not allocated. Press F1 (Help) for online help
4. Enter the required options in the VSAM Options field. Press F1 (Help) for online help.
5. To action the parameter group, press F6 (Action).
6. To update the parameter group, press F3 (File).

To deallocate the EASIUDB data set

1. Enter **/PARMS** on any panel.
2. Enter **U** next to the $AC EASIUDB parameter group in the Files category.
3. Delete the text in the Dataset Name field. When this text is deleted, the data set is deallocated (if already allocated). Press F1 (Help) for online help.
4. To action the parameter group, press F6 (Action).
5. To update the parameter group, press F3 (File).

EASIUSER application definitions, group definitions, and user definitions are stored in the EASIUDB data set.

To maintain EASIUSER definitions

1. Enter **/ACADMIN.E** on any panel. The User Easinet Definitions panel displays.
2. Select an option from the panel. Press F1 (Help) for online help.

More information:

Implementing EASINET (see page 43)

How You Implement and Administer MAI

To implement and administer MAI, perform the following tasks:

- Specifying the CA SOLVE:Access Database (ACDB)
- Specifying External Application LU Names (EXTAPPLPOOLS)
- Specifying MAI Parameters (MAIPARMS)
- Defining and Authorizing Users
- Defining and Maintaining Sessions

To use the Session Replay Facility (SRF), also define the MAI Session Trace File.

If you have a SNA performance product and want to correlate CA SOLVE:Access virtual sessions with the real VTAM session, perform one of the following tasks:

- For CA NetSpy, implement the NetSpy Session RTM Interface.
- For the IBM Tivoli NetView Performance Monitor (NPM), implement the NetView Synergy Interface (NSI).

CA SOLVE:Access Database (ACDB)

The ACDB parameter group defines the CA SOLVE:Access definition database, which is used to contain MAI session definitions and screen images.

Note: For information about using this parameter group, see the *Installation Guide*.

External Application LU Names (EXTAPPLPOOLS)

The EXTAPPLPOOLS parameter group allows you to specify LU name prefixes to use with MAI-FS and MAI-OC sessions.

Note: For more information about using this parameter group, see the *Installation Guide*.

Specify MAI Parameters (MAIPARMS)

The MAIPARMS parameter group is used to customize the following MAI parameters:

- Session skip characteristics, such as the skip keys and skip characters.
- MAI Menu title.
- MAI Session controls, such as session end panel format, data stream save option, and VTAM response option.

To customize these parameters

1. Enter **/PARMS** on any panel.
2. Enter **U** next to the $AC MAIPARMS parameter group in the tuning category.
3. Change the fields as required. Press F1 (Help) for online help.

   This parameter group has three pages. Use the F10 and F11 keys to move between the pages.
4. To action the parameter group, press F6 (Action).
5. To update the parameter group, press F3 (File).

Define and Authorize Users

CA SOLVE:Access users must be authorized to use the MAI-FS feature. This authorization is assigned by modifying their user ID definition.

If your system is using the UAMS security system, do the following for each user that is to have MAI-FS access authority.

Note: You must have UAMS authority to perform the steps.

To authorize users

1. Enter **/UAMS** on any panel.

   The UAMS : Primary Menu displays.
2. Select the option to add or update a user ID (option A or U).
3. Go to the Access Authorities panel, and enter **Y** for Access Services.
4. Go to the MAI Details panel, and enter the required values. Press F1 for online help.
5. To update the details, press F3 (File).

More information:

Authorize MAI-FS User IDs (see page 70)

MAI Sessions

Each MAI session has various operational characteristics including a session ID and logon string. When new sessions are added, they inherit default values that can be customized. Session details are stored on the CA SOLVE:Access database (ACDB) and can be maintained using MAI Session Definition Maintenance (MSDM).

Set MAI Session Defaults

Each MAI-FS session has certain operational characteristics such as the logon string and a set of jump keys. Users with the appropriate authority can define new sessions from the following:

- MAI Stored Definition Maintenance (MSDM)
- MAI Menu

When a new session is defined, it assumes default session characteristics. These default characteristics are maintained using the MAI Session Defaults option on the SNA Access Services Administration Menu.

To maintain these default characteristics

1. Enter **/ACADMIN.D** on any panel.

   The MSDM : Default Session Details panel displays.

2. Change the fields as required. Press F1 (Help) for online help.
3. Press F3 (File) to save the changes and exit.

Maintain MAI Session Lists

The list of MAI-FS sessions that are available to a user can be maintained using the MAI Session Definition Maintenance (MSDM) facility. These definitions exist on the following levels:

Session Definitions

A session definition contains data (session characteristics) that MAI-FS uses for setting up a session with another application. Session definitions are stored in a session list.

Session Lists

A session list contains a number of session definitions. Each MAI-FS user is associated with a named Session List.

To access the MSDM function, use the **/MSDM** shortcut on any panel.

The MSDM : Primary Menu displays.

More information:

Advanced MAI-FS Customizing (see page 143)

Define the Session Trace File (TRFILE)

The Session Trace File (TRFILE) is a VSAM KSDS that is used as a repository for the Session Replay Facility (SRF). This parameter group is used to define the TRFILE data set if you intend to use SRF.

To allocate the TRFILE

1. Enter **/PARMS** on any panel.
2. Enter **U** next to the $AC TRFILE parameter group in the Files category.
3. Enter the data set name in Dataset Name field. If you do not specify the data set name, the file is not allocated. Press F1 (Help) for online help.
4. Enter the required options in the VSAM Options field. Press F1 (Help) for online help.
5. To action the parameter group, press F6 (Action).
6. To update the parameter group, press F3 (File).

To deallocate the TRFILE data set

1. Enter **/PARMS** on any panel.
2. Enter **U** next to the $AC TRFILE parameter group in the Files category.
3. Delete the text in the Dataset Name field. When this text is deleted, the data set is deallocated (if already allocated). Press F1 (Help) for online help.
4. To action the parameter group, press F6 (Action).
5. To update the parameter group, press F3 (File).

More information:

Session Replay Facility (see page 161)

Generate Logon Scripts

When a session is started, there is usually a standard dialog to effect a logon to the application. Optionally, the dialog can also include initial menu options to navigate to a specific panel within the application. For example, an application defined to run under TSO can have a logon that could include:

- Logon to TSO with user ID and password entry
- Welcome broadcasts
- Application selection

The logon to an application can be automated using a session script, an NCL procedure that uses MAI functions to send and receive data streams. A session script can be generated using a recording function where a user logs on to the application in real time, captures the data streams, converts the required operations and data to NCL statements, and saves the procedure in a partitioned data set for subsequent use.

To access the Logon script generation function, enter **/ACSGEN** on any panel.

The Logon Recording : Application Selection list displays.

More information:

Generating Logon Scripts (see page 133)

Implement the NetSpy Session RTM Interface (NETSPYRTM)

NetSpy Session RTM Management is a component of CA NetSpy that allows correlation between virtual sessions, such as CA SOLVE:Access (MAI full-screen) sessions, and the underlying real session.

The NETSPYRTM parameter group allows you to implement the interface to CA NetSpy by specifying the Session Manager Name, that is, the name by which your CA SOLVE:Access region is defined to CA NetSpy.

To implement the interface

1. Enter **/PARMS** on any panel.
2. Enter **U** next to the $AC NETSPYRTM parameter group.
3. Tab to the Session Manager Name Field and enter the required value. Press F1 (Help) for online help.
4. To action the parameter group, press F6 (Action).
5. To update the parameter group, press F3 (File).

Customizing the NetView Synergy Interface

The NetView Synergy Interface (NSI) allows you to monitor sessions which are connected to MAI, using IBM Tivoli NetView Performance Monitor (NPM). The IBM *NPM Installation and Customization Guide* describes NPM functions.

To access NSI support (see page 195), enter **/ACADMIN.I** on any panel.

Terminal Security

You can set terminals that are logged on to an application to time out after a period of inactivity. This time-out reduces the security risk of having them logged on but unattended.

A general time-out facility is provided in your product region. Use this facility to specify time-out intervals and actions for all terminals. The time-out facility identifies a user at a terminal as having been inactive for a period and executes an action. Possible actions are to ring the terminal alarm, or to log the user off the system.

Note: For more information about the time-out facility, see the *Reference Guide*.

Support for Generic Resources

For information about how to implement support for generic resources, see the *Installation Guide*.

Chapter 5: Implementing EASINET

This section contains the following topics:

EASINET

EASINET provides an easy logon procedure and a guide to the applications that are available to users. Terminals placed under EASINET control can be defined to VTAM with the CA SOLVE:Access APPL name as their controlling application. This definition is achieved either on the terminal definition statements by including the LOGAPPL parameter, or by using the VARY LOGON command. VTAM then logs the terminal on to CA SOLVE:Access. As soon as a terminal is connected to CA SOLVE:Access, it is handed to the EASINET feature.

The use of NCL procedures gives you complete flexibility when designing screen panels, and provides logic capability, access to files, and access to security system information. EASINET provides a simple way to customize the presentation of the network to users.

NCL can also be used to implement applications at the network boundary. Corporate logon systems and online help systems can all be implemented from the EASINET procedure.

You require familiarity with NCL and its use when implementing applications with full-screen panels. Details of these facilities are provided in the *Managed Object Development Services Programmer and Administrator Guide*.

Sample EASINET Systems

CA SOLVE:Access is distributed with two working samples of an EASINET system. They are the $EASINET NCL procedure and a suite of NCL procedures and panels that make up the EASIUSER application. Alternatively, you can develop your own EASINET system using the guidelines described later.

EASINET Procedure

The $EASINET procedure provides an enterprise-wide logon function and consists of the following components:

- The $EASILOGON, $EASIHELP, $EASIACCEPT panels in the panels database
- The $EASINET NCL procedure
- DEFLOGON and APPLSTAT commands in the distributed CA SOLVE:Access initialization procedure, $ACINIT.

Note: The name of the CA SOLVE:Access initialization procedure is specified in the ACINIT Customizer parameter group.

The $EASINET procedure performs the following functions for full-screen devices:

- Displays a logo panel containing terminal name, date, time, and a function key grid showing which function keys effect a logon to each application, plus the status of each application. Various applications have been included, some of which may not be available in your system. The panel includes three input fields; a user ID field, a password field, and a command field.
- Press a Function key (other than F7 or F19) to log on and pass the user ID and password fields entered as logon user data in TSO format.
- Press F7 or F19 to log on and pass the user ID and password fields entered as logon user data in NetMaster format.
- Press F1 or F13 to display another full-screen panel showing the online help.
- Press Enter with no data in the command field to redisplay the panel.
- Press Enter and enter HELP in the command field to display another full-screen panel of online help.
- Press Enter with anything other than HELP in the command field, to attempt a logon with the characters entered.
- If an online help panel display times out (that is, the INWAIT time period for that panel expires after one minute), redisplay the main logo panel.

Note: The procedure uses the command field if you press Enter, and the user ID and password fields if you press a function key.

For LU1 devices, the procedure performs the following functions:

- Sends a salutation message, followed by any current broadcast messages
- Prompts you for input
- If the first word input is HELP, sends a help message, and prompts again
- If the first word input is TSO, prompts for a user ID and password, then attempts a logon passing this information as logon user data
- Any other input attempts a logon using the full text entered

The distributed EASINET procedure ($EASINET) includes extensive documentation describing how various steps are performed. $EASINET can be found in the ACTEXEC library. Review the procedure to understand how to implement an EASINET system.

To use the $EASINET procedure, specify $EASINET as your EASINET procedure name (see page 47).

EASIUSER Application

The EASIUSER suite of procedures and panels perform the following functions for full-screen devices:

- Display a logo panel containing terminal name, date, time, and NMID. This panel includes the following input fields:
  - User ID
  - Password
  - Group
  - Command
- Display a customized list of applications for selection after a user ID and password have been entered and validated. The format of the list can be provided in the following ways:
  - A function key grid showing which function keys effect a logon to each application
  - A numbered selection list
- You can effect a logon to each application by typing the selection number in the command input field, or by pressing the corresponding function key. The selection list can also display a description of the applications listed. The status of each application displays on both options.
- A UDB (EASIUDB) controls the customized list presentation. The UDB is opened during initialization.
- You can create a set of MAI definitions from the customized application screen. MAI definitions can be created from all the applications listed, or only those applications that you select.
- Allow the user to change their password, or force them to change it when it expires.
- Display a nominated panel.
- Display a panel to describe how EASIUSER works for the end user.
- Provide an extensive tutorial for use of EASIUSER ($EASITUT) for the administrator.
- Invoke the security system to determine a user's attributes and validate their password.
- Assign the customized application list by user ID, name, or generic user ID. The application can assign the list by LUNAME with minor changes.

- Allow the user to select which customized list to display from the first panel.
- Allow the user to be passed directly from the first screen to a selected application, bypassing the selection list or grid completely.
- Allow or disallow the use of the command line to enter applications of commands not presented on the customized panel.
- Allow the passing of user data to the selected application in various formats (UID PSW or UID/PSW or UID PSW COMMAND).
- Pass the user to CA SOLVE:Access, and pass the initial command to execute.

The EASIUSER procedures provide the same functionality for LU1 devices as the $EASINET procedure.

The EASIUSER procedures ($EASIUSR, $EASIUP1, $EASIUP2, $EASITUT, $EASIUP5, $EASIUP6, $EASIMAI, and $EASIDSC) can be found in the ACTEXEC library. They have extensive comments which describe how the suite of procedures works together.

We recommend that you run the procedure $EASITUT from OCS to review the tutorial on how to use EASIUSER. The customized application lists are kept on a user database (UDB) that is specified using the EASIUDB Customizer parameter group.

To use the EASIUSER system, specify $EASIUSR as your EASINET procedure name (see page 47).

Select the EASINET System

To select your EASINET system

1. Enter **/PARMS** on any panel.
2. Enter **U** next to the $AC ACINIT parameter group in the Names category.
3. Enter the name of the EASINET Procedure. To use the sample systems, enter the values indicated in the previous sections. Press F1 (Help) for online help.
4. To action the parameter group, press F6 (Action).
5. To update the parameter group, press F3 (File).

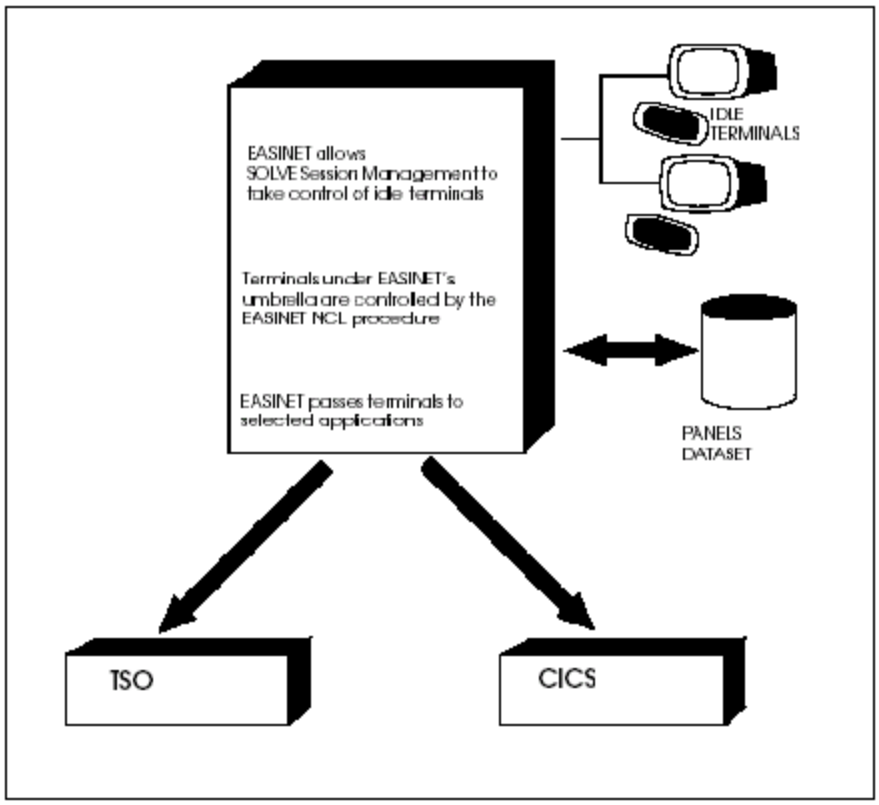
EASINET NCL Procedure

The NCL procedure you nominate using the ACINIT parameter group is run each time a terminal is connected to CA SOLVE:Access. The procedure then determines the actions to take with that terminal. Typically, the procedure performs the following actions:

- For a full-screen device such as a 3270, displays an initial panel (for example, a logo or corporate identity panel), possibly containing instructions for its use. For a line-by-line device (LU1 terminal), writes salutation messages to prompt for user input.
- Waits for user input.
- Checks the input. This input could be a special command the procedure recognizes (such as HELP), a request to enter a tutorial of some sort, or a request to LOGON to another application.
- If necessary, displays further panels or messages.
- For a request to LOGON, requests EASINET to pass the terminal to the requested application program, such as TSO, CICS, or IMS.

The NCL procedure uses the &PANEL statement to display full-screen panels that have been predefined through the SOLVE online editor. Use the &WRITE and &PROMPT statements to communicate with LU1 terminals. These statements and the editor are described in the *Network Control Language Reference Guide*.

The &LOGON statement (see page 53) requests EASINET to pass a terminal to another application. This statement is shown in the following illustration.



The EASINET NCL procedure can make full use of NCL facilities. For example, it can:

- Determine the node name for the terminal connected using the &LUNAME system variable, and display different initial logos or messages for different user terminals.
- Determine the type of terminal using the &ZLUTYPE system variable.
- Determine the size of the terminal using the &LUROWS system variable and display panels of varying depths accordingly.
- Use the full logic and arithmetic capabilities of NCL. For example, set up user and global variables, access VSAM databases.

The EASINET NCL procedure provides a friendly and easy-to-use means for end users to log on to applications, or lets them write powerful applications using a high-level, easy-to-use language.

Logon user data can also be passed to an EASINET procedure, and made available in the variables &1, &2, &3, and so on, when the procedure starts.

More information:

Logon User Data Formats (see page 95)

EASINET Panels and Messages

The EASINET NCL procedure controls the terminals under EASINET. Each terminal can be regarded as running its own copy of the procedure.

On a full-screen device such as those devices in the 3270 family, the procedure can display any number of full-screen panels you have previously defined using the SOLVE online panel editor. The content of these panels varies according to your installation, but the following system-maintained variables are useful:

&LUNAME

Provides the node name of the terminal.

&TIME

Provides the time (updated each time the panel is written).

&DATEn

Provides the date in various formats.

&ZREMIPA

Provides the remote IP address on TN3270 connections (only available if CA NetMaster NM for TCP/IP is installed).

Any other information the EASINET procedure sets up can be displayed on a panel. The procedure assigns a value into a user or global variable, and the panel is defined to contain the variable name. The current contents of the variable replace the names when the panel displays.

The EASINET procedure can also support line-by-line terminals, referred to as LU1 terminals. For example:

- IBM 3767 terminals, or any terminal supported as an SNA LU1 terminal through use of a protocol convertor or appropriate NCP software.
- ASCII printing terminals.

The procedure can send messages to LU1 terminals using the standard &WRITE statement with its options for carriage return, line feed, and form feed functions. The &PROMPT verb is used to read input from the device. Once again, system variables can be used in messages sent to the terminal. Data sent to and received from the terminal can be handled in a transparent fashion, so that any hexadecimal character can be sent to or received from the terminal. EASINET handles devices that are not true terminals, such as intelligent dial-out modems defined to the network as LU1 devices.

Broadcast Messages

CA SOLVE:Access includes a facility for broadcasting several lines of data to all or selected terminals under EASINET control. This broadcast is effected from the Broadcast Services : Primary Menu, option B, or by using the NSBRO commands, as detailed in the *Command Reference*.

For full-screen devices, the broadcast is sent if a panel containing one or more of the following variables is currently displayed:

- &BROLINE1 - broadcast line 1
- &BROLINE2 - broadcast line 2
- &BROLINE3 - broadcast line 3
- &BROLINE4 - broadcast line 4

These variables can be included anywhere in a panel definition, and should be defined as output-only fields. Each variable should be the only one in a field that occupies the whole line because each broadcast line can be up to 78 characters long. The broadcast automatically re-displays a panel, without EASINET procedure intervention. The variables are replaced by the text you broadcast.

For LU1 devices, the broadcast is automatically sent when set, regardless of the current device status. The EASINET procedure can also send broadcasts to LU1 devices using &WRITE statements.

Alternative Broadcast Methods

The special &BROLINEn variables are defined so that when any are changed, every panel displaying the changed variable is refreshed:

- Automatically, if it is under EASINET control
- The next time the panel displays, for users logged on to CA SOLVE:Access

It is possible that the &BROLINE variables do not provide sufficient text space for some broadcast applications. To automatically redisplay up to one screen of information, you can enter information into the conventional user or global variables on a panel, and select the panel services #OPT BCAST option from the panel definition for broadcasting.

The #OPT BCAST option lets you specify that the panel is to be redisplayed automatically whenever any of the &BROLINE variables are changed, even if the panel does not include any of the &BROLINE variables.

This facility lets you trigger an automatic refresh of a panel by changing the value of any of the &BROLINE variables, while using your own choice of variables to contain the panel display text.

Data Stream Compression

Displaying logo screens is inefficient. They typically include long strings of identical characters.

CA SOLVE:Access automatically compresses data streams to optimize line utilization and minimize the impact of swamping the network at start-up time, or during error recovery processing.

&LOGON Statement

EASINET can require a terminal it is controlling to be passed to another application. The decision for this operation depends on input received from the terminal and its interpretation in the procedure.

The NCL &LOGON verb is used to direct EASINET to pass a terminal to another application.

The &LOGON statement has the following formats:

```
&LOGON panelname deflogon-path-name text
```

or

```
&LOGON APPL=applname [ LOGMODE=logmode ] panelname text
```

Symbolic Application Selection

If the &LOGON statement is coded specifying a *deflogon-path-name*, the EASINET procedure does not know the application name explicitly.

A *deflogon-path-name* is selected according to data received from the terminal, but the actual application selected depends upon the definition of the DEFLOGON path itself.

Example:

```
&LOGON OKPANEL F02
```

OKPANEL

Specifies the name of an NCL screen panel required by the *panelname* parameter of the statement.

F02

Specifies the name of a DEFLOGON path defined in the DEFLOGON table.

This form of the &LOGON verb is the most flexible. DEFLOGON definitions can be changed dynamically at any time and the EASINET procedure automatically reacts to the changes without needing modification.

More information:

DEFLOGON Table (see page 82)

Explicit Application Selection

If the &LOGON statement is coded specifying the APPL=applname, the applname operand of the keyword specifies explicitly the network name of the target application. For example:

```
&LOGON APPL=TSO &LOGON APPL=&APPL1
```

For the applname, an optional VTAM logmode table entry name can be nominated to supply suggested BIND parameters for the session between the terminal and the selected application.

This form of the &LOGON statement is useful for handling one off requests, where the application has not previously been defined by any deflogon path.

Acknowledgment Panel

In both forms of the &LOGON statement, the panelname parameter nominates an NCL panel definition which displays briefly on the terminal before it is passed to the selected application. This facility allows a brief acknowledgment screen to be presented to the user indicating their selection has been accepted.

The length of time this acknowledgement panel is presented depends upon the INWAIT parameter specified on the panel definition. This should be a small value, perhaps 0.2 seconds.

Use the UNLOCK=NO option on the panel definition so that the terminal keyboard is not unlocked when the message displays.

If the terminal is an LU1 device, the panelname parameter (although still required by the syntax of the &LOGON statement) is ignored because panel display does not apply on these terminals. To implement an acknowledgment message (for example, REQUEST ACCEPTED) on an LU1 device, assign the required text to the &SYSMSG variable before executing the &LOGON statement. If the logon is successful, the text assigned to &SYSMSG is sent to the terminal before it is passed on.

Logon User Data

Some applications accept logon user data associated with the connection of a terminal to the application. User data typically consists of initial parameter information such as user ID, password, or initial transaction name, depending upon the individual application. Because &LOGON logs the terminal on to the selected application, the &LOGON statement lets you specify any user data that you require to be passed to the application with the terminal. The user data can be explicit. For example:

```
&LOGON REQACCEPT TSO USER01
&LOGON APPL=TSO OKPANEL USER01
```

or can be derived from the user's input:

```
&LOGON REQACCEPT &CMD &UID/&PW
```

The variables &CMD, &UID, and &PW can contain data the user typed into three input fields on a panel, or entered in response to prompts on an LU1 terminal. They are substituted into the &LOGON statement when it is executed. If the logon is successful, the acknowledgment panel that appears for a short time is named REQACCEPT.

&LOGON Statement Completion

If EASINET determines that terminal control cannot be passed, control is returned to the next statement after the &LOGON statement. In this case, the &SYSMSG variable contains an error message which explains the reason for the unsuccessful attempt. The procedure normally re-writes a panel containing the &SYSMSG variable so the user can see the error, perhaps after modifying the text of the error message.

On an LU1 terminal, &WRITE is used to send the contents of &SYSMSG to the terminal.

If the terminal is passed to another application successfully, the EASINET procedure (or rather the copy of the procedure operating for this terminal) is automatically terminated by EASINET.

It is possible, because of VTAM error conditions, for a CLSDST pass NOTIFY to never be received, and so the &LOGON statement will never complete. If, in such circumstances, the original terminal ends its session with the third party application, and returns to CA SOLVE:Access, the error condition will be detected. The logon process will be failed with &SYSMSG set to N22731 TERMINAL LOGGED BACK ON WITH NOTIFY OUTSTANDING. Such problems could be temporarily circumvented by running with SYSPARMS VTAMNTFY=NO or by specifying NOTIFY=NO on the DEFLOGON command.

Function Keys in EASINET

All NCL procedures support the use of function keys on full screen devices. When an NCL procedure regains control after an &PANEL statement, it means *one* of the following conditions:

- A key has been pressed to transmit input.
- The INWAIT time defined on the panel has expired.

On returning from the &PANEL statement, the &INKEY variable is set to indicate the key used to cause the input. Each key on the terminal that can cause input is given a name, and &INKEY is set to the name of the appropriate function key. The NCL procedure (in this case, EASINET) can tell which key was pressed, so specific processing can be performed for each key.

The &INKEY variable can have the following values:

Key used	&INKEY value
ENTER	ENTER
F1	PF01
F2	PF02
...	...
F24	PF24
PA1	PA1
PA2	PA2
PA3	PA3
None	null value

Note: PA key interception applies only if the &CONTROL PAKEYS option is active for the procedure.

Function Key Support Extension

Function key support can also be extended into the DEFLOGON entry matching mechanism. When the EASINET procedure issues an &LOGON statement, it can use the variable &INKEY to indicate the key pressed. DEFLOGON table entries can provide information such as application program required and the source of logon user data.

For example, if the EASINET procedure issues the following statement:

```
&LOGON REQACCEPT &INKEY &UID/&PW
```

and the user entered their TSO user ID and password and pressed F2, the statement would read (after variable substitution):

```
&LOGON REQACCEPT PF02 MYUSERID/MYPASS
```

If the following DEFLOGON command had been entered:

```
DEFLOGON INPUT=PF02 APPL=TSO DATA=INPUT1
```

then the user would be logged on to TSO, and their user ID and password supplied as logon user data.

The full range of 24 function keys is supported. Thus it is possible to format a logo as a menu showing the function key users press to select a desired application. All logon user data options are also available.

Note: Issuing the type of &LOGON statement in the example is normally meaningless when a user presses Enter. The EASINET procedure decides (using an &IF statement) that a function key has been utilized before issuing this format of &LOGON statement.

More information:

DEFLOGON Table Entries (see page 84)

Command Execution in the EASINET Procedure

Most SOLVE and VTAM commands can be executed from the EASINET procedure, which is considered to be a user with a command authority of 255. However, all commands should be executed from an internal environment using an &INTCMD statement, and the results of those commands retrieved using the &INTREAD statement (as for any full-screen NCL procedure).

If any commands are executed natively, the results of the commands are queued pending the completion of the procedure. If the procedure ends because a successful &LOGON statement is executed to pass control to another application, these results are purged. However, if the &LOGON causes a logon to this system or the procedure terminates for any other reason, then panels with the results of the commands are displayed. Acknowledge these results by pressing Enter, before proceeding. The same applies to any &WRITE TERM=YES statements (the default) issued, or if &CONTROL TRACE is in effect, or if NCLTRACE TERM is active.

On an LU1 device, most commands can be executed natively, and responses are sent to the terminal automatically.

EASINET and Network Security

The EASINET procedure interacts directly with the security subsystem. That subsystem can be either the UAMS component or a security exit provided by the installation. Using &SECCALL statements allows EASINET to perform the following functions:

- Validate user ID/password combinations.
- Force a user to change an expired password.
- Determine the privileges of a user.
- Perform other validation or inquiry functions required by the security exit.

&SECCALL allows unlimited flexibility in any dialog between EASINET and the security exit, so extensive security validation can be performed at the front end of the network. Users can be presented only with information pertinent to them. For example, EASINET can determine which applications a user is allowed to log on to, and then present those applications for selection.

Note: For a description of the &SECCALL NCL verb, see the *Network Control Language Reference Guide*.

EASINET and the Remote Operator Facility (ROF)

The Remote Operator Facility (ROF) is a facility of CA SOLVE:Access that allows an operator to sign on to a remote location, execute commands, and have the results returned. The EASINET procedure can use ROF signons to other systems if it issues, for example, the following command:

```
&INTCMD SIGNON NM2
```

This creates a session with the NM2 region. The user ID, on whose behalf the session is created, is a special one made up of the local system user prefix (from the NMSUP JCL parameter) and the suffix EASI (for example, NM1EASI). This user ID must be defined in the remote SOLVE region for the signon to proceed.

Following signon, the EASINET procedure can use:

```
&INTCMD ROUTE NM2 command string
```

to send commands to the other region, and receive the results using &INTREAD statements.

This facility can be useful when implementing a centralized logon capability, where user ID information for all staff is held at a central location. Any EASINET system can send requests for verification or alteration to a central system, so you do not have to have multiple user ID databases.

How You Troubleshoot the EASINET Procedure

You could encounter errors in the NCL procedure as it develops. An error encountered when operating from a full-screen device, terminates the procedure, clears the screen, and displays error messages on a panel. Press Enter to acknowledge any errors. The procedure is then rerun.

Before you acknowledge an error, you have an opportunity to change the procedure, provided it is not preloaded. For an LU1 device, error messages are sent to the terminal and the procedure rerun, without any user intervention.

Use the &CONTROL TRACE or &CONTROL TRACELOG options in a procedure to help find a problem. Remember that if &CONTROL TRACE is used, a full screen device does not display output from the trace until the procedure terminates. Using &WRITE LOG=YES TERM=NO to write specific information to the log can also help. The use of the NCLTRACE command can also be helpful.

Activate the Procedure

Once thoroughly tested, the EASINET NCL procedure you have written must be activated.

To activate the procedure

1. Update the ACINIT parameter group.
2. Specify the name of the new procedure.
3. Action the parameter group.

   By default, the EASINET procedure is loaded into memory and a single copy shared between all terminals. When you action the parameter group, the old copy is unloaded. If your EASINET procedure name is not changed, you should still action the parameter group to unload the previous copy of the procedure.

EASINET Coding Recommendations

This section gives some tips on the most effective ways of coding an EASINET NCL procedure, and provides ideas on how EASINET may be used.

The following points help make EASINET more efficient:

- Avoid setting up local variables before displaying a panel (at least the initial logo panel), or executing a &PROMPT. If local variables are used, nullify them before displaying a panel or executing &PROMPT. Each variable set up by the EASINET procedure occupies a certain amount of storage. If the procedure sets up a local variable before displaying the initial panel, the storage for the variable is duplicated for each terminal and sets up the variable in all cases.
- Consider using global variables. Only one copy of a given global variable occurs in the entire system.
- If the EASINET procedure uses &FILE statements, ensure the &FILE CLOSE statement is issued before the procedure is suspended for any length of time (for example, across the display of a panel or a prompt). This frees the storage associated with file processing.
- Consider using the SYSPARMS MAXPANEL command to ensure the main EASINET panels are maintained in storage to eliminate high activity on the panels data set.
- Always display the main EASINET panel from the top level EASINET procedure. Otherwise, extra storage is used for each terminal to record the procedure nesting information.
- Ensure that all panels used by the EASINET procedure, and any other procedures it can call, are available using the PANELS library path.

EASINET Termination

Ensure that procedure termination does not result in uncontrolled loops. Procedures are rerun unless:

- The session is terminated using the CLSDST * command.
- The session is terminated with &LOGON.
- A logon to CA SOLVE:Access is initiated.

EASINET Extensions

The following points may give you some ideas:

- An operator can send messages to a particular terminal using the standard broadcast facilities (for a specific terminal broadcast). The EASINET procedure can let a terminal user return a message to the operator. For example, there could be a command field on the panel into which the user can enter:

  ```
  MSG I WILL BE LEAVING IN 5 MINS.
  ```

  The EASINET procedure recognizes the MSG command. It executes the MSG command using &INTCMD to transmit the message to a central operator, and indicates to the sender if the message was sent successfully.

- DEFLOGON commands defining applications that you can select can contain a line of descriptive text. Use the SHOW DEFLOGON command to extract the application description and display the descriptions of the various applications on the EASINET panel.

- A full news service can be implemented using NCL file processing techniques. Use a UDB with any amount of news information using a User Services NCL procedure. The User Services procedure lets an operator update news information, and the EASINET procedure can display database records to users on request. Simplify the EASINET procedure by saving the news information in global variables (at system startup and in the User Services procedure). These global variables need to be included in an EASINET panel.

- Because EASINET acts as a gateway through which network users must pass to reach network applications, extensive security checking can be implemented. Typical implementations are:

  - Enforce entry of user ID and password from the EASINET screen, then use &SECCALL CHECK to verify the password is correct.
  - Use &SECCALL GET to list the applications a user is entitled to access, and present a menu with only those applications and their current status.
  - This approach is widely used to validate users attempting to log on from dial-in terminals.

Systems Programmer's Summary

The following EASINET checklist may be useful for Systems Programmers:

- Specify the names of the CA SOLVE:Access initialization procedure and the EASINET procedure in the ACINIT Customizer parameter group.

  Note: The default CA SOLVE:Access initialization procedure is $ACINIT and is copied to the TESTEXEC library during installation and set-up.

- Define at least one initial logo panel, using the online editor.
- Code an NCL EASINET procedure and place it in your SOLVE procedure library. The name of this member must be that identified in the ACINIT parameter group. Use the distributed procedure $EASINET as an example.
- Code the necessary DEFLOGON commands used to support the &LOGON statements issued by your EASINET procedure in the $ACINIT member.
- Read the description of the SYSPARMS command PANLBUFF operand in the *Reference Guide*. The PANLBUFF operand lets you change the maximum buffer usage allowed for concurrent terminal output operations.

  The default setting, which allows for up to 40 storage pages to be used at any one time, is sufficient for good performance in almost all cases. If you find it takes too long for all the EASINET terminals in a large network to receive their initial display panel during network startup, use the SYSPARMS PANLBUFF operand to increase the buffer allocation. Use increments of five buffers until the startup time is satisfactory, or until no further improvement is achieved.

  Note: At this time, you should add the SYSPARM command to your CA SOLVE:Access initialization procedure.

  This operand also governs the rate at which general broadcasts are propagated through the EASINET network. If too few buffers are available the time required for a broadcast to reach all terminals will increase.

  The default setting is adequate in almost all cases and should not be changed if system performance is acceptable.

If there is no performance increase when you raise a buffer limit, revert to the default setting. This means that output propagation is being throttled by factors outside the control of CA SOLVE:Access (for example, line speeds).

How EASINET Handles Different Screen Sizes

Various models of terminals in the 3270 range have differing screen sizes. For example:

Model 2

80col X 24row

Model 3

80col X 24row or 80col X 32row

Model 4

80col X 24row or 80col X 43row

Model 5

80col X 24row or 132col

A terminal under EASINET control operates in its largest 80 column mode. For instance, models 3 and 4 operate as 80 columns by 32 rows, and 80 columns by 43 rows respectively, but model 5 operates as 80 columns by 24 rows (the model 5 is swapped to its larger dimension when logged on to CA SOLVE:Access).

The EASINET procedure determines the size of the screen it is controlling from the &LUROWS system variable. This variable contains a number that represents the number of screen rows, for example, 24 or 32. The procedure can then display a panel specially set up to display on that particular size of screen. If a panel has more lines than can fit on the terminal, the panel display is truncated.

CA SOLVE:Access supports any screen size up to 255 columns 255 rows.

Note: For more information about panel definition, see the *Network Control Language Programming Guide*.

How EASINET Reacquires Terminals

CA SOLVE:Access attempts to reacquire terminals that have been under EASINET control but disconnected for various lost terminal conditions. These attempts include using the SYSREQ key to enter conditional or unconditional logoff from EASINET.

Note: CA SOLVE:Access does not attempt to reacquire failed Telnet LUs.

In addition, if EASINET passes a terminal to another application and fails, it attempts to reacquire the terminal. If this reacquisition is successful, control returns to the EASINET procedure after the &LOGON statement with an appropriate error message in &SYSMSG. The EASINET procedure is only terminated if the new session is established successfully or EASINET is unable to reacquire the terminal.

If reacquisition does fail, the terminal reverts to the default VTAM USS table control. (Define your default USS table so that the terminal is always logged on to CA SOLVE:Access again whatever input is received from the terminal.)

Any reacquisition attempts are logged.

Activity Log Messages

Whenever a terminal passes from EASINET to another application, a message is written to the SOLVE activity log detailing the name of the terminal, and the application to which it has been passed. If the SYSPARMS command CONMSG=YES operand has also been entered, a message is written to the log each time a terminal is connected to CA SOLVE:Access. All errors encountered are written to the log, and the EASINET procedure can write additional messages using the &WRITE LOG=YES TERM=NO statement.

Chapter 6: Implementing MAI

This section contains the following topics:

About MAI

The Multiple Application Interface (MAI) is a feature of CA SOLVE:Access that allows a user at a single terminal to operate more than one session at the same time. The MAI feature also provides other functions, for example, MAI can be used to record anything that displays on a terminal's screen.

MAI operates in either of the following modes:

- Full Screen mode: MAI-FS
- Operator Control (line-by-line) mode: MAI-OC

The principal difference between the two modes is their display on the screen; MAI-FS uses the entire screen and MAI-OC uses one line at a time.

Note: Full Screen mode (MAI-FS) sessions are described here. For information about MAI-OC sessions, see the *Reference Guide*.

Initialization Processing

All initialization processing for the MAI component of CA SOLVE:Access is performed by Customizer (see page 29).

MAI-FS Defaults

A number of parameters determine the presentation of MAI to users of the system, and provide for flexible implementation of the product. All these parameters have default settings that can be changed to suit your system's requirements.

Parameters are set using Customizer parameter groups or CA SOLVE:Access Administration options.

More information:

Specify MAI Parameters (MAIPARMS) (see page 36)
Set MAI Session Defaults (see page 37)

VTAM APPL Definitions

When MAI establishes a session with a target application on behalf of a SOLVE region user, a new VTAM access method control block (ACB) is opened, unless an ACB that is already open can be shared. The *applname* used is the name of a predefined VTAM application, defined to VTAM with an APPL statement.

MAI uses this ACB to start a session with the selected application (in which MAI emulates an LU Type 0, LU Type 1, or LU Type 2 device). The application behaves as if it is communicating with a terminal whose node name is the *applname* of the VTAM ACB.

Throughout this guide, the term LU name is used when referring to the APPL used by MAI to establish a session with its target application.

The use of MAI therefore requires the definition of a set of VTAM APPL statements.

LU Names

The user can specify the LU name used by CA SOLVE:Access to establish the session directly when the session is created. Alternatively, CA SOLVE:Access can choose the next available in a series. The name is of the format *xxxxxnnnn*, where *xxxxx* is any one- to five-character string and *nnnn* is a three- or four-digit number in the range 001 through 9999. The string *xxxxx* is the MAI prefix. For MAI-OC sessions the default value of this string is NMMAV, which can be changed by using the SYSPARMS MAIOPREF=*xxxxx* command.

Note: The maximum length for an LU name is eight characters. If a five-character prefix is used, the numeric suffix is restricted to three digits, that is, in the range 001 through 999.

Each MAI prefix is considered to be a pool and there are different prefixes for MAI-FS and MAI-OC pools. These pools are defined using the EXTAPPLPOOLS parameter group in Customizer (see page 35).

For MAI-FS sessions, pools can be defined as part of an application definition using the DEFLOGON MPREF=*xxxxx* operand.

More information:

DEFLOGON Table Entries (see page 84)
LU Name From a Pool (see page 77)

MAI Installation Exits

You can use exits to validate and/or change the characteristics of an MAI-FS session, collect accounting statistics for an MAI session or send data streams to devices when session jumping is performed. The ways these exits can be used are detailed in the appendices.

Session Scripts

The use of session scripts can significantly improve the operation of MAI-FS sessions, especially when used to automate session start.

Logon scripts can be generated using a recording facility.

More information:

MAI-FS Session Scripts (see page 121)
Generating Logon Scripts (see page 133)

Authorize MAI-FS User IDs

CA SOLVE:Access users must be authorized to use the MAI-FS feature. This authorization is assigned by modifying their user ID definition.

If your system is using the UAMS security system, perform the following procedure for each user that is to have MAI-FS access authority.

To authorize a user for MAI-FS in UAMS

1. Enter **/UAMS** on any panel.

   The UAMS : Primary Menu displays.
2. Select the option to add or update a user ID (option A for Add User Definition or U for Update User Definition).
3. Go to the Access Authorities panel, and enter Y next to Access Services.
4. Go to the MAI Details panel, and enter the required values. Press F1 for online help.
5. To update the details, press F3 (File).

Note: You must have UAMS authority to perform these steps.

If your system is using a full security exit, the authorities are defined using the following structured fields (SF):

- X'0027'—Access Services authority
- X'0200'—Privilege Class
- X'0201'—Session Model
- X'0202'—A and E Commands
- X'0203'—Active Session Limit
- X'0204'—MSDM Access

Note: For a description of these structured fields, see the *Security Guide*.

Privilege Class

Privilege Class determines the functions available to a user and the way in which MAI-FS is presented to the user. Valid values are:

A

This allows the user access to all MAI-FS facilities. The MAI : Primary Menu displays on entry to MAI. No restrictions are imposed on the user's establishment of sessions or the MAI-FS commands that the user may enter.

B

The range of MAI-FS commands available to the user is restricted as described below. The MAI : Primary Menu displays on entry to MAI.

The user can establish only those sessions that are listed on their MAI : Primary Menu.

C

As for class B, but the sessions listed on the MAI : Primary Menu are started automatically on entry to MAI and the screen is assigned to the first session in the list. The MAI : Primary Menu will appear as the user jumps around the session circle.

D

As for class C, but the MAI : Primary Menu is eliminated from the session circle.

Note: You must also define a session model for users with privilege classes B, C, or D.

Command Privilege

Control of the MAI-FS environment and the manner by which sessions are selected is achieved through a range of MAI-FS commands. There are three categories of MAI commands. They are:

- Primary commands
- Line commands
- Session commands

The privilege class assigned to each MAI user (A, B, C, or D) controls which MAI commands are available to the user. Users with Privilege Class A can enter all supported commands in all categories. Users with Privilege Class B, C or D can enter commands in all categories with the following exceptions:

- Prohibited primary commands: L and U
- Prohibited line commands: C, D, L, M, P, R, and U
- Prohibited session command: C

Restricting command privileges limits users to a pre-defined set of sessions whose attributes they cannot change. This is a technique for controlling general access to MAI-FS facilities.

Note: For a description of how to use MAI commands, see the *User Guide*.

Session Model

Predefined or stored session definitions allows frequently established sessions to be created using parameters stored by MAI-FS on behalf of the user. The user does not have to enter those parameters each time that session is required.

Stored definitions are maintained on a user ID basis and can be created on behalf of a user ID by being logged on as that user ID. Stored definitions can also be copied from one user ID to another using the MAI Session Definition Maintenance (MSDM) feature (see page 103).

Note: For information about how to create a session definition, see the *User Guide*.

To create user IDs with associated stored definitions without having to set up stored definitions for those user IDs, you can designate a session model in the MAI-FS user ID definition.

The Session Model field allows the entry of a one- to eight-character model user ID. The user ID being defined inherits the stored definitions associated with the model user ID. When users with a session model enter MAI-FS, they are presented with a menu selection list made up of all the stored definitions that exist for the model user ID. An asterisk (*) in the field indicates that the user's own stored definitions is used. This setting is useful when modeling user IDs. It allows users to be added with model MAI attributes, but having individual MAI session profiles.

Users who are modeled cannot change the model's definitions in any way, nor can they add new stored definitions (regardless of privilege class).

Note: The selection of stored definitions associated with the model can be set up by logging on using the model user ID, entering MAI-FS and using the U primary command. This process and an alternative method using MSDM, is described in detail in the *User Guide*.

Use of a session model eliminates the need to set up stored definitions for each user ID.

The session model field is mandatory for users with privilege classes B, C, or D.

A and E Primary Commands

The A and E primary commands allow the user to activate all the session definitions associated with the user ID and are essentially equivalent to placing the S (Select) line command against every inactive session displayed on the MAI : Primary Menu. The only difference between the A and E commands is that E causes activation of all sessions then eliminates the MAI menu from the session circle.

If the user ID is to have several stored definitions available (perhaps in a model user ID), but is likely to need some of those sessions infrequently, you might want to prevent the user from starting sessions except by individual selection.

This field allows (enter a Y for Yes) or disallows (enter an N for No) use of the A and E primary commands. If disallowed, the user can activate sessions only by individual selection using the S line command.

Active Session Limit

This field determines whether a user is limited in the number of sessions that they can have active at the same time. Possible values are 0-255. A value of 0 indicates no limit applies. This is the default when this field is omitted.

How You Start MAI-FS Sessions

MAI-FS users who have privilege classes B, C, or D, start MAI-FS sessions by selecting the session from the MAI-FS menu. This selection establishes the session using the options provided in the associated stored session definition.

Class A users can set up sessions for which they have no stored definition and modify the attributes of their existing sessions. If a user sets up a new session, the user can be presented with the Logon Details panel, which requests information required for starting the new session.

Regardless of the origin of the session request information, MAI requires the same information before a session can be started.

The Logon Details panel is described in the *User Guide*. The following information can be provided by the user, automatically loaded from a predefined definition, or it can in some cases be allowed to default:

- A logon request which supplies the following:
  - The name of the application with which the session is required
  - Logon user data to pass to those applications that support such data (for example, user ID and password)
- A description of the session (optional).
- A session ID to identify the session.
- A node name that overrides the automatic selection of an LU name by MAI-FS. The target application sees a logon from this LU name.
- Any function keys, PA keys, or both, that the user wishes to allocate to this session. These keys provide a means of jumping away from the session to other MAI-FS sessions or CA SOLVE:Access functions.
- Various optimization parameters.
- The name of a LOGMODE table entry to use to suggest SNA session parameters for the session.
- The name of an NCL procedure to perform scripting functions for the session (see page 121). This procedure can automatically drive the session, simulating keyboard input and monitoring output.

The MAISESS command can also be used to start and stop predefined MAI-FS sessions. This command allows MAI-FS session control without requiring the use of the MAI menu. Nonterminal owning regions can manage MAI-FS sessions using the MAISESS command. This feature allows background regions, such as BMON and BLOG, and ROF users to start and control MAI-FS sessions. Using the MAISESS command is described in the *User Guide*.

Logon Request

The user's logon request is a character string entered from the Logon Details panel. This string first goes through a process of variable substitution, whereby words beginning with an ampersand (&) have values substituted in their place.

The logon request can contain any NCL system or global variables, including the &USERID and &USERPW variables. Variables are of most benefit when used within stored logon definitions.

After substitution, the string is matched against the DEFLOGON entry table to determine the application with which to create the session and pass any logon user data.

More information:

DEFLOGON Table (see page 82)
How EASINET Uses the DEFLOGON Table (see page 83)
How MAI Uses the DEFLOGON Table (see page 83)

Session Description

The logon request string can be concatenated with, or coincide with, a session description string. This description appears to the right of the Status field on the MAI : Primary Menu entry for the session.

A separation character must be provided at the end of the logon request so that MAI can recognize the beginning of the session description. The separation character is programmable by the MAIDSSEP system parameter. If the separation character is the first character of the field, all succeeding characters are taken as both the logon request and the session description.

As is the case for the logon request string, the session description can contain NCL system and global variables, and goes through a process of variable substitution before being displayed. If no session description is provided, any text specified using the DESC= operand of the DEFLOGON entry used by the session is displayed.

Selection of LU Name for an MAI-FS Session

Before MAI can request a session to be started with a target application the system must allocate the LU name that will be used to act as the terminal end of the session. The allocation of LU name can be allowed to default, or a specific LU name can be designated.

LU Name From a Pool

If no specific LU name is required, MAI-FS generates one. The generated name consists of the prefix (set by the EXTAPPLPOOLS Customizer parameter group or DEFLOGON) followed by a suffix in the range 001 through 9999. The number chosen is the first number not already in use for another MAI-FS session for this user or another user. The maximum length of the constructed LU name is eight characters. The prefix can be one through five characters and the suffix can be three or four digits. Alternatively, MAI-FS can share a name already in use.

Specific LU Name

If the user requires an MAI-FS session with a target application in which the MAI-FS LU name must be the name of a specific terminal, the NODE NAME field on the Logon Details panel allows the user to specify the LU name that MAI-FS is to use.

This technique requires the user to know the terminal to be used on the session, but it also means that the identity of the terminal is predictable.

This facility is necessary for establishing an MAI-FS session which has to have particular attributes. For example:

- An IMS system is generated with its IMS Master Terminal (primary operating console) having an LU name of MTO3270P.
- A CA SOLVE:Access operator needs an MAI-FS session with IMS to act as the Master Terminal Operator.
- The operator requests an MAI-FS session specifying MTO3270P as the LU name to use on the session.

Because the MAI default process never generates the LU name of the IMS Master Terminal, the only way in which MAI could be used to drive the MTO session is by specifying the LU name explicitly.

An MAI Installation Exit (see page 69), (MAIEX02), if provided, is run whenever a session request is about to be processed. This exit can override the LU name or prefix if necessary.

How Terminals Are Locked or Disconnected

CA SOLVE:Access provides facilities to lock a terminal in such a way that you must enter your password before processing can recommence. The disconnect facility of CA SOLVE:Access allows you to terminate the session between MAI and the user's terminal.

A terminal can be locked or disconnected as a result of a user request (using entry of a command in the MAI Selection Menu) or as a result of a time-out condition (see page 41).

Session control options control the availability of LOCK and DISC as MAI menu commands and session commands. These options are set using the MAIPARMS parameter group in Customizer (see page 36). The options (Lock Allowed? and Disconnection Allowed?) are on Page 3 of the panel.

The LOCK, DISCONN, RECONN, and CANCEL commands are described in the online help.

Chapter 7: Defining and Administering Applications

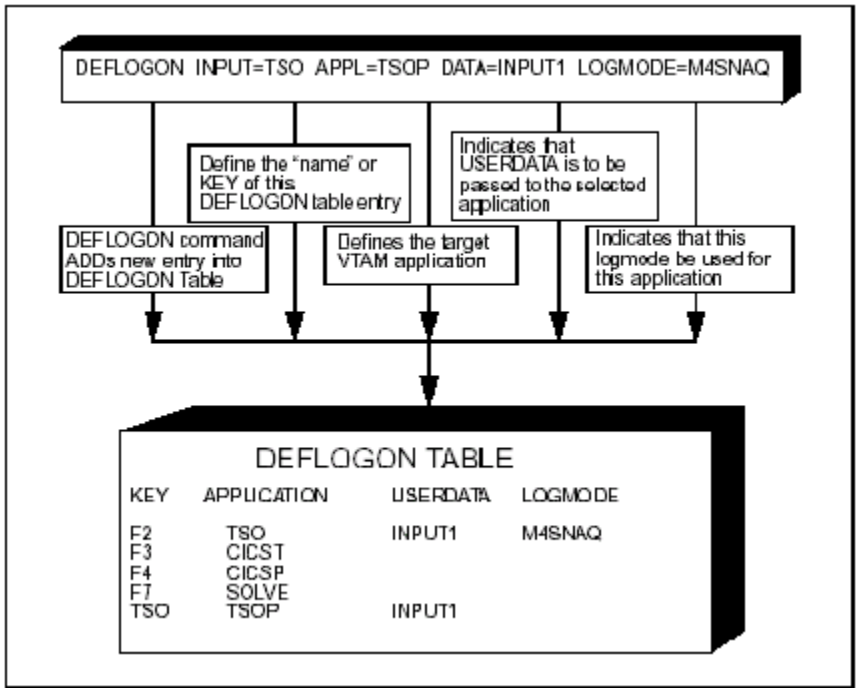
This section contains the following topics:

DEFLOGON Table

The EASINET and MAI-FS components of CA SOLVE:Access are used to access other (VTAM) applications such as TSO, CICS, and IMS. To enable access to these applications, you define logon paths that identify the target applications and operational parameters. These logon paths are defined in the DEFLOGON table.

The execution of DEFLOGON commands (see page 84) defines a set of DEFLOGON entries in the DEFLOGON table. These commands are typically defined in your CA SOLVE:Access Initialization Procedure. The name of this procedure is specified in the ACINIT parameter group of Customizer with a default name of $ACINIT. DEFLOGON commands can also be entered at any time using OCS.

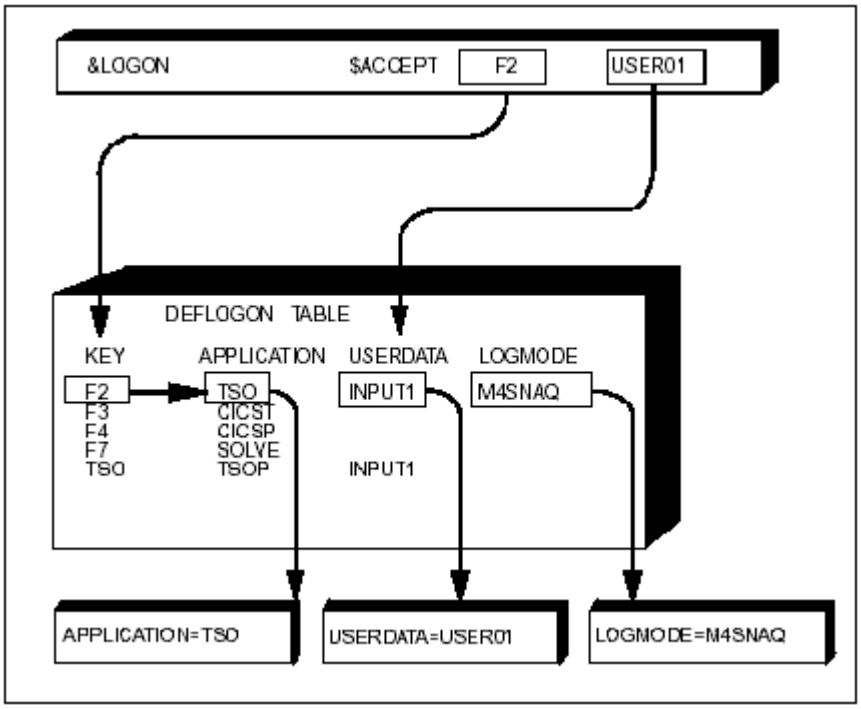
An input word, which is typically a logical application name (for example, CICSTEST), identifies each DEFLOGON table entry. The table entry contains the real VTAM application name and can include other operational parameters such as user data format and VTAM Logmode name (see the following illustration).



When searching the DEFLOGON table, the search is either generic, in which case a partial match is acceptable, or equal, in which case the input word must exactly match a DEFLOGON key. The choice of search is specified on the DEFLOGON command.

How EASINET Uses the DEFLOGON Table

EASINET uses the DEFLOGON table when the &LOGON statement does not include the APPL= operand. In this case, the third word of the statement (after the &LOGON and panel-name, as delimited by blanks) is used as a search string. The string is matched against the entries in the DEFLOGON table to determine to which application the terminal is passed, and whether to pass any user data (see the following illustration).



Some applications, such as TSO and SOLVE, support the provision of certain data at logon time, such as user ID and password information.

How MAI Uses the DEFLOGON Table

The table is used by MAI when starting a session. The first word in the MAI session definition's Logon String is the input word that is matched against the DEFLOGON table with remaining data passed to the application as user data if the DEFLOGON entry permits this.

DEFLOGON Table Entries

The DEFLOGON command adds an entry to the DEFLOGON entry table. Both EASINET and MAI use this table. The key of each entry is a one- to eight-character string, referred to as the DEFLOGON path name. Other information associated with each entry is:

- The name of the application to which the entry applies.
- The user data, or source of the user data, to be passed to the application.
- A LOGMODE table entry used when the terminal is passed.
- Whether INQUIRE APPSTAT is used to determine if an application is active.
- Whether EASINET uses a VTAM third party notify when passing a terminal to the application.
- A textual description of the application. This displays on MAI selection menus.
- An MAI-FS session ID, which is the default if the user has not provided a session ID.
- Whether the session attempts to share an already-open VTAM ACB.
- A description of the logon path, which is the default session description for any MAI session using this logon path.
- Whether MAI will respond to the application before or after the data has been sent to the terminal.
- The prefix name used by MAI as the logical terminal pool.
- The name of the application's refresh key to be displayed as part of the lost screen image message.
- The MAI inactivity time in minutes, which is the time between an application output that unlocks the keyboard, and a response from the terminal or script.
- The action to take when an MAI session expires.
- The number of sessions allowed.
- Whether a PassTicket is generated.
- A script to use for this application.

DEFLOGON Command—Define Logon Information

The DEFLOGON command defines the logon information required to support a target application for the EASINET and MAI facilities.

The command has the format:

```
DEFLOGON { INPUT=text | IN=text }
         [ IGE|IEQ ]
         [ APPL={ vtamappl |*}]
         [ DATA={ userdata | INPUT | INPUT1 | NOEASINET } ]
         [ LOGMODE=table_entry ]
         [ MID=mai_id ]
         [ MSHR={YES | NO } ]
         [ APPSTAT={ YES | NO } ]
         [ MAXSESS=0 | number ]
         [ MRESP={ AFTER | BEFORE | PLU } ]
         [ MPREF=prefix ]
         [ MTLIMIT=minutes ]
         [ MTACTION={ CANCEL | CC | SCRIPTSTART |
                      SCRIPTEND | EVENT } ]
         [ NOTIFY={ YES | NO } ]
         [ PASSTKT={ NO | YES | TSO | value } |
           PTKT={ NO | YES | TSO | value } ]
         [ REFRESH=key ]
         [ SCRIPT='procedure_and_parameters']
         [ DESC=session_description ]
```

INPUT=*text*

Specifies the input string entered by the operator, or the leading characters of the string to be entered.

IGE | IEQ

Identifies the type of match to make against the input string. IGE specifies that a logon string matches on the leading characters of the logon path defined using the INPUT operand. If you want an exact match, specify IEQ.

Default: IGE

APPL={ *vtamappl* | * }

Specifies the VTAM APPL name of target application or * for CA SOLVE:Access' own APPL.

Default: An application name matching the INPUT text

DATA={ *userdata* | INPUT | INPUT1 | NOEASINET }

Specifies the user data to pass to the target application. Specify INPUT if you want to pass the text as entered by the user, or INPUT1 if you want to pass the text as entered minus the first word.

Specify NOEASINET if the target application is another SOLVE region and you want to bypass its EASINET facility.

If the *userdata* format is used and the data contains blanks, the string must be enclosed in quotes (”).

LOGMODE=*table_entry*

Specifies the VTAM logmode table entry to use when the session is established.

MID=*mai_id*

Specifies the default MAI session ID if one is not supplied during creation of an MAI-FS session. If not supplied, the APPL name is used as the default.

MSHR={ YES | NO }

Determines whether MAI sessions created using this entry use ACB sharing.

APPSTAT={ YES | NO }

Determines whether the application status is queried before establishing a session.

MAXSESS={ 0 | 9999 }

Specifies the maximum number of active sessions with the application.

Range: 0 to 9999

Default: 0 (no maximum)

MRESP={ AFTER | BEFORE | PLU }

Specifies whether MAI-FS sessions using this definition are to respond to the application before or after the data has been sent to the terminal.

- AFTER provides synchronous session control, with the terminal's responses being echoed to the application.
- BEFORE specifies that MAI-FS always responds positively upon receipt of data. The setting also provides an asynchronous mode of operation.
- PLU specifies that the response mode of the application is used on the physical sends. Terminal responses are echoed to the application.

Default: AFTER

MPREF=*prefix*

Specifies an ACB prefix for MAI to use as a logical terminal pool.

MTLIMIT=*minutes*

Specifies a time-out limit that applies to MAI sessions established using this DEFLOGON entry.

MTACTION={ CANCEL | CC | SCRIPTSTART | SCRIPTEND | EVENT }

Specifies the action to take when an MAI session is inactive for the time specified by MTLIMIT.

- CANCEL cancels a session.
- CC causes a conditional cancel.
- SCRIPTSTART initiates a session script in SKIP mode.
- SCRIPTEND initiates a session script in END mode.
- EVENT issues an event.

NOTIFY={ YES | NO }

Specifies whether EASINET uses the VTAM Third Party Notify capability when establishing a session.

Default: Set by the VTAMNTFY system parameter

PASSTKT={ NO | YES | TSO | *value* }

Specifies whether to generate a PassTicket for use by MAI-FS. The ticket is available through the &USERPW system variable.

- NO does not generate a PassTicket.
- YES generates a PassTicket for the application determined by the APPL operand.
- TSO generates a PassTicket for the local Time Sharing Option (TSO), TSO*smf_sys_id*.

  smf_sys_id is the local System Management Facilities (SMF) system ID.
- *value* generates a PassTicket for the specified application or TSO.

Default: NO

REFRESH=key

Supplies the name of the application's refresh key. MAI-FS includes this key as part of the lost screen image message (N54201).

SCRIPT='*procedure_and_parameters*'

Specifies the name of an NCL procedure and parameters to be invoked as a session script when the session is started. The first word is the procedure name (maximum 8 characters), and subsequent words are parameters that are passed to the procedure as arguments (&1, &2, and so on). If the operand includes parameters, it must be enclosed in quotes.

Limits: 26 characters

DESC=*session_description*

Enter a text description of the application identified by this DEFLOGON entry. If specified, DESC must be the last operand. The value can consist of any characters (including blanks) without the need for enclosing the data in quotes. If your system is configured with the MAI feature, this description is displayed next to the MAI menu entry for this application.

How Logons Are Processed

When a logon request is processed, the first word of the request is used to search the DEFLOGON table. The rules governing which entries match the logon request depend on which of the IGE or IEQ operands was specified on the DEFLOGON command used to define the entries. If IGE was specified (or defaulted), the entries are searched for the best available match according to the following rules:

- Each DEFLOGON entry is checked to see if its key (value in the INPUT operand) matches the corresponding number of characters of the first word of the logon request, working from left to right.
- If an entry is found that exactly matches the logon request (that is, in length and content), then this entry is chosen. If no entry exactly matches the request, then the entry with a key that matches the request for the longest number of characters and which was not specified with the IEQ operand is chosen.
- If entries match for an equal number of characters, the entry with the longest key is chosen.
- If no DEFLOGON entries are found to match the logon request, the session request is rejected with an appropriate error message.

In the following example, three DEFLOGON commands have been defined:

```
DEFLOGON INPUT=A APPL=TSO
DEFLOGON INPUT=AB APPL=IMS
DEFLOGON INPUT=ABC APPL=CICS
```

If the text AB1234 makes up the first word of the logon request, it is interpreted as a request to pass the terminal to IMS.

Although the complete text does not match any of the three definitions, a match is found when the first two definitions are compared against the string. The first definition (the key is A, that is, only one character) matches the first character of the text. The second definition (the key is AB, that is, two characters) matches the first two characters of the text. No match is found for the third definition because its three-character key ABC does not match the first three characters of the text.

The choice between the first and second definitions is resolved by selecting the definition with the longest key, INPUT=AB.

This method of selection allows the use of a generic DEFLOGON command by systems where, for example, all TSO user IDs start with the same letter.

Entries defined with the IEQ operand specified are searched for an exact match only. The first word of the logon request must be the same as the value specified in the key of the DEFLOGON entry for the match to be successful. If no DEFLOGON entries exactly match the logon request, the session request is rejected and an appropriate error message is generated.

Personalized Logons

The DEFLOGON command can be used to define personalized logons for users. For instance, a TSO user with a TSO user ID of TS431 wants to use a different size and logon procedure operand for each TSO session. Three separate DEFLOGON definitions could be defined for this user. For example:

```
DEFLOGON INPUT=TS4311 APPL=TSO DATA=TS431
DEFLOGON INPUT=TS4312 APPL=TSO DATA=TS431 SIZE(3000)

DEFLOGON INPUT=TS4313 APPL=TSO DATA=TS431 PR(PROC2)
```

In these examples, the user can enter TS4311, TS4312, or TS4313 as a logon request. One of the DEFLOGON entries would be matched and a logon to TSO instigated, passing the TSO user ID and other information dependent upon the user's input.

DATA=INPUT Definition

The following DEFLOGON command causes CA SOLVE:Access to pass all the data that makes up the logon request to the selected application as logon user data:

```
DEFLOGON INPUT=U APPL=TSO DATA=INPUT
```

This technique can be used to provide a simple method of TSO logon where a user can enter their TSO user ID, and identify that they want a TSO session and provide their TSO user ID simultaneously.

As an example, a user whose TSO user ID is USER01 might enter the following as a logon request:

```
USER01 SIZE(3000) PR(PROC1)
```

As the logon request starts with the letter U, CA SOLVE:Access will match it with the following definition:

```
DEFLOGON INPUT=U
```

CA SOLVE:Access determines from the definition that this represents a request to pass the terminal to TSO and also to pass as user data to TSO the character string:

```
USER01 SIZE(3000) PR(PROC1)
```

This is the entire logon request. TSO will use this string to identify USER01 as the user logging on and will therefore not have to prompt for the user ID and can immediately request that the user's TSO password be entered.

DATA=INPUT1 Operand

Another form of DATA=INPUT processing enables all but the first word of the logon request to be passed as logon user data to the application. This option is invoked using the DATA=INPUT1 operand. For example, coding the command:

```
DEFLOGON INPUT=TSO APPL=TSO DATA=INPUT1
```

would enable a user to logon to TSO by entering the string:

```
TSO USER01 SIZE(3000)
```

The first word (TSO) is stripped and the rest, (USER01 SIZE(3000)) is passed to TSO as user data, identifying the user ID and other logon information.

REPLOGON Command—Modify Logon Definition

Existing DEFLOGON definitions can be modified using the REPLOGON command by an operator who has the appropriate authority level. The REPLOGON command has the same format and operands as the DEFLOGON command (see page 85).

The REPLOGON command can also be used to add a DEFLOGON table entry. If only the INPUT operand is specified, then the DEFLOGON entry that matches the text in the input operand, displays in the command line ready for modification. When Enter is pressed, the updated entry replaces the existing entry.

For example, if an operator wanted to change the APPL= operand of DEFLOGON definition USER01, they enter the following command at the OCS command line:

```
REPLOGON INPUT=USER01
```

The current definition of USER01 appears on the command line, for example:

```
REPLOGON INPUT=USER01 APPL=TSO DATA=INPUT MID=TSO
```

The APPL name could then be changed as required. Press Enter to update the definition.

Note: The value of the INPUT= operand can be modified too because a REPLOGON command adds a new definition if it does not exist.

DELLOGON Command—Delete Logon Definition

DEFLOGON entries which are no longer required can be deleted from the definition table by an OCS operator of the appropriate authority level, using the DELLOGON command.

The command has the following format:

```
DELLOGON { INPUT=text | APPL=applname }
```

INPUT=*text*

Deletes only the entry specified from the table.

APPL=*applname*

All entries that provide a logon to application *applname* are deleted from the table.

Note: If *text* and *applname* are coded as ALL, all DEFLOGON entries are deleted.

SUSLOGON Command—Suspend Logon Path

You might want to suspend, temporarily, the availability of a DEFLOGON entry, for example, if the application is not available and will not be available for some time. In this case, the SUSLOGON command can be used to disable selected DEFLOGON entries or all entries referring to a particular application. In addition, the command can define a message that appears at the user's terminal if a logon request matches the suspended entry.

The command has the following format:

```
SUSLOGON { INPUT=entryname | APPL=applname }[ TEXT=msgtext ]
```

INPUT=*entryname*

Specifies the name of a defined DEFLOGON entry to suspend. Defined entries can be displayed using the SHOW DEFLOGON command. Use this operand if you want to suspend only a single entry. Users who attempt to log on to an application by entering the text that matches *entryname* are denied access. INPUT= can be abbreviated to IN=.

APPL=applname

Specifies the application to which logons are suspended. Multiple DEFLOGON entries can exist that allow users to log on to an application in different ways. This operand causes suspension of all such entries.

TEXT=msgtext

Specifies the text of a message to display at the user's terminal if they attempt to log on to the suspended application. The text can be up to 50 characters in length, and must be enclosed in single quotes (') if it contains any blanks. If the TEXT operand is not entered CA SOLVE:Access displays the text:

```
N20213 LOGON TO SELECTED APPLICATION IS SUSPENDED.
```

ACTLOGON Command—Reactivating Logon Path

DEFLOGON entries that were previously suspended by SUSLOGON commands can be reactivated using the ACTLOGON command.

The command has the following format:

```
ACTLOGON { INPUT=entryname | APPL=applname }
```

INPUT=*entryname*

Specifies the entry name of a defined DEFLOGON entry to activate. Defined entries can be displayed using the SHOW DEFLOGON command. Previously suspended entries are displayed in high intensity. Use this operand if you want to activate a single entry.

APPL=*applname*

Allows logons to the specified application. Multiple entries can exist that allow users to log on to an application in different ways. This operand causes activation of all such entries.

Permanent Changes

Changes made online using the DEFLOGON, REPLOGON, or DELLOGON commands are effective only until CA SOLVE:Access is restarted. Permanent changes can be made by updating the commands supplied in the CA SOLVE:Access initialization procedure, which is in the TESTEXEC data set. The name of the CA SOLVE:Access initialization procedure is specified in the ACINIT Customizer parameter group. The default procedure name is $ACINIT.

Access to CA SOLVE:Access from an EASINET Terminal

Terminals running under the control of EASINET are, in fact, in session with the SOLVE region. However, to gain access to CA SOLVE:Access itself (by supplying your user ID and password), you must code a DEFLOGON statement to define CA SOLVE:Access as a valid application.

Two methods are available:

- Use a DEFLOGON statement to define the SOLVE APPL name explicitly. Here are two examples:

  ```
  DEFLOGON INPUT=NM APPL=NMPROD

  DEFLOGON INPUT=PF04 APPL=NMPROD (so F4 can be used)
  ```

- Code a special value of the APPL operand:

  ```
  DEFLOGON INPUT=NM APPL=*
  ```

  APPL=* indicates that the name of the SOLVE APPL is substituted in this definition. This value allows a standard DEFLOGON statement to be included in all CA SOLVE:Access initialization procedures, without having to be customized to the particular system concerned. The name of the CA SOLVE:Access initialization procedure is specified in the ACINIT Customizer parameter group. The default procedure name is $ACINIT.

Note: The APPL=* definition identifies the system controlling the terminal. If a second system is running, a logon to that system requires a proper DEFLOGON statement explicitly defining that system's APPL name.

Logon User Data Formats

CA SOLVE:Access supports the following formats of logon user data:

- NOEASINET
- EASINET
- User ID password menu-option

This data can be supplied using an external source, such as a VTAM USS table, or using a combination of DEFLOGON commands and &LOGON statements.

NOEASINET Format

If you supply logon user data when a terminal connects to CA SOLVE:Access and that user data is the string NOEASINET, CA SOLVE:Access bypasses the EASINET function for that terminal. The EASINET procedure is not executed on behalf of the connected terminal. When the user logs off, the terminal is disconnected from CA SOLVE:Access and not retained under EASINET control.

This user data is most often used when a terminal under the control of an EASINET in one domain, logs on to CA SOLVE:Access in another domain. You might not wish to use EASINET in that domain, but to log on directly to CA SOLVE:Access. In addition, when you log off the other system, your terminal is not retained by that system's EASINET, as would normally be the case.

Another possible use of NOEASINET is where a logon to CA SOLVE:Access is effected directly from VTAM USS facilities, for example, and EASINET is to be bypassed. If NOEASINET is supplied as user data when logging on to CA SOLVE:Access from that EASINET, the data is ignored and the terminal is retained by EASINET when the session terminates.

EASINET Format

If EASINET is the first word of the logon user data supplied when a terminal connects to CA SOLVE:Access, then all following words are tokenized and placed into the variables &1, &2, &3, &4 and so on, before invoking the EASINET procedure. The procedure can then interrogate the variables in the normal way.

This user data is most often used when one EASINET procedure controlling a terminal requires a function performed by an EASINET procedure in another SOLVE region. It can log the terminal onto the other system using &LOGON, and pass special information using the logon user data.

User ID Password and Menu Selection

EASINET can provide CA SOLVE:Access with a user ID, password, and an initial menu selection or shortcut using logon user data. This allows you to bypass the SOLVE logon panel and possibly some menus. The following rules apply:

- The user ID must be supplied first, followed by one or more blanks and then the password.
- The user ID cannot be supplied without the password.
- The password can be followed by one or more blanks, followed by an initial SOLVE menu selection or shortcut (for example, M.TSO or /BCAST). Providing a menu selection is optional.
- If the above rules are violated, the user data is ignored and the normal SOLVE logon panel displays.
- If the user ID or password is invalid, the normal SOLVE logon panel displays with an error message.

Providing user data in this way implies the NOEASINET option, so EASINET is bypassed. However, where a logon is effected from EASINET to its own SOLVE region, the terminal returns to EASINET control when the session ends.

The LOGONUSRDATA parameter group of Customizer determines whether providing the user ID and password in this manner is valid, and defaults to NO.

Override EASINET/NOEASINET

EASINET and NOEASINET can be overridden using the Logon Data Interpretation field in the ACINIT parameter group. The values for this field are:

BYPASSOK

Specifies that the EASINET procedure is always called unless the first word of the logon string is NOEASINET. Specifically, where the first word of the logon string is:

EASINET

Processing is as described.

NOEASINET

Processing is as described.

Other

The word EASINET is assumed. The EASINET panel displays, and any data is tokenized and passed to the EASINET (in &1, &2, and so on) where it can be used as required.

NOBYPASS

Specifies that the EASINET procedure is always used, regardless of the first word of the logon string. The EASINET procedure cannot be bypassed using NOEASINET. Specifically, where the first word of the logon string is:

EASINET

Processing is as described.

NOEASINET

The EASINET process is not bypassed and processing is as if the first word is EASINET.

Other

Processing is as for BYPASSOK.

To specify an alternative option

1. Enter **/PARMS** on any panel.
2. Enter **U** next to the $AC ACINIT parameter group in the Names category.
3. Enter the required value in the Logon Data Interpretation field. Press F1 (Help) for online help.
4. To action the parameter group, press F6 (Action).
5. To update the parameter group, press F3 (File).

Access to EASINET from TSO and TSS

The TSO interface component of the EIP option lets a TSO user connect to CA SOLVE:Access directly from a TSO session. By default, a TSO logon of this type bypasses EASINET and passes you directly to SOLVE itself.

If a TSO user wants access to EASINET, or any NCL applications operating under EASINET, code the keyword EASINET on the NMLOGON command, followed by any parameter data to pass to the EASINET procedure. For example:

```
NMLOGON APPL(NMT) EASINET(PHONES)
```

In this example, the NMLOGON command establishes a session with CA SOLVE:Access on behalf of a TSO user, and the EASINET procedure then runs the session.

EASINET can tell that it is a TSO user from the &ZTSOUSER system variable. The variables &1 and &2 are set to the name of the TSO user ID and the word PHONES respectively. This facility lets TSO users access NCL applications resident under EASINET (such as an internal telephone list) without having to have CA SOLVE:Access user IDs defined for them.

Application Status Monitoring

CA SOLVE:Access can be configured so that terminal users see the status of each application they can log on to in the following ways:

- From any desired EASINET panel.
- Using &WRITE statements on an LU1 terminal.

Specify the Applications to Monitor

CA SOLVE:Access asks VTAM for the status of a set of predefined applications at a predefined frequency.

To specify the applications to monitor, specify the network names of the applications using the APPLSTAT APPL=*xxxx* command in the CA SOLVE:Access initialization procedure.

Note: The name of the CA SOLVE:Access initialization procedure is specified in the ACINIT Customizer parameter group. The default procedure name is $ACINIT.

Examples:

```
APPLSTAT APPL=IMS
APPLSTAT APPL=TSOP
```

Set the Application Monitoring Frequency

By default, CA SOLVE:Access requests the status of these applications every minute.

To change this frequency, issue the following command to specify an alternative frequency in seconds (minimum 30):

```
SYSPARMS APPLSTIV=180
```

Specify any APPLSTIV operand before APPLSTAT commands.

Global Variables

The status of each application is conveyed in a global variable with the format:

&GLBL*applname*

applname is the name of the application concerned. For example, if the application TSOP is being monitored, a global variable named &GLBLTSOP is created, which contains the status of application TSOP.

Use these global variable names on EASINET panels or &WRITE statements.

The various status values and their meanings are as follows:

ACTIVE

The application is active and logons enabled.

INACTIVE

The application is not active.

NOLOGONS

The application is active but not accepting logons.

UNKNOWN

The application is not known to VTAM.

NOTAPPL

The name specified is not an application.

UNAVAIL

Unable to determine status, or status inquiry in progress

The global variables created are available for any NCL procedure to interrogate, and are directly included within a full screen panel. If an NCL procedure deletes an application status global variable (using an assignment statement to assign a null value), the monitoring of the application stops until another APPLSTAT command is entered for it. A procedure can temporarily assign alternate text into one of these global variables before displaying them on a panel. The value of the variable is overwritten the next time the application status is monitored.

The PURGE APPLSTAT command can also be used to terminate specific application monitoring.

Use the SYSPARMS APPTXT*n* commands to change the default status text values.

How You Can Ensure the Completion of Application Status Requests

The CA SOLVE:Access request for the status of an application can take some time to complete, especially if the application is cross-domain (on another host). In some systems, it is necessary to delay the CA SOLVE:Access initialization procedure slightly after the APPLSTAT commands have been processed (using the &DELAY statement), so that all application status requests are complete before any terminal is connected. (Terminals cannot connect to CA SOLVE:Access until the CA SOLVE:Access initialization procedure has been processed). A delay of one or two seconds is sufficient; the maximum is 30 seconds. If the status of an application has not been determined, its global variable is assigned a value of UNAVAIL.

Note: The name of the CA SOLVE:Access initialization procedure is specified in the ACINIT Customizer parameter group. The default procedure name is $ACINIT.

Use the NCL &APPSTAT built-in function to determine the status of an application. If a cross-domain application status inquiry takes longer than 30 seconds to complete, the &APPSTAT function returns a not available status. &APPSTAT is used when the application status at this instant is required.

Chapter 8: MAI Session Definition Maintenance

This section contains the following topics:

Administration of MAI-FS Definitions

MAI Session Definition Maintenance (MSDM) is used to administer MAI-FS definitions.

MAI definitions exist on two levels:

Session Definitions

A Session definition contains data (session characteristics) that MAI-FS uses for setting up a session with another application. Session definitions are stored in a session list.

Session Lists

A session list contains a number of session definitions. Each MAI-FS user is associated with a named session list.

Access MSDM

To access MSDM, use either of the following ways:

- Enter **/MSDM** at the SOLVE : Primary Menu ===> prompt.
- From the UAMS : Primary Menu, select option M - MAI Session Definition Maintenance, or, using panel skipping, enter **A.AC.MS** on the Primary Menu.

Note: You must have authority for System Support in UAMS to update other users. For more information about UAMS, see the *Security Guide*.

The MSDM : Primary Menu displays with the following options:

A - Add a Session List

Add a new MAI session list for a user ID. When this option is selected, the Session Definition panel is presented. Enter the information about the initial session definition for the new session list.

B - Browse Session List

Browse the MAI session list for a user ID. This option displays a selection list of MAI session definitions, from which session definitions can be browsed.

U - Update Session List

Update the MAI session list for a user ID. This option displays a selection list of MAI session definitions.

The listed session definitions can be browsed, updated, copied, inserted, moved, reset, or deleted.

D - Delete Session List

Delete the MAI session list for a User ID. The Delete Session List panel is presented. You are prompted to confirm or cancel the deletion.

L - List Session Lists

View a selection list of session lists. The listed session lists can be browsed, updated, deleted, or copied or session definitions can be copied. To restrict the list, specify a generic prefix in the Userid field. For example, enter **USER** to limit the list to user IDs starting with the letters USER.

C - Copy Session List

Create a user MAI session list based on an existing session list. For example, copy the USER1 session list to create a session list for USER2 with the same session definitions.

CS - Copy Session Definitions

Copy complete MAI session definitions from one user to one or more users with existing MAI session lists. Specify the source of the copy in the Userid field. You are then prompted for the target User IDs and copy options. A new session list is created in the MAI stored definition file with a complete copy of the MAI session definitions from the source user ID.

You can copy one or a number of MAI session definitions (session IDs) from a source user ID to one or more target user IDs. You can specify the target user IDs (session lists) by selecting an ID from a selection list, or by specifying a user ID mask or full user ID.

Sessions can either be copied and replaced, or copied only if they meet certain match criteria.

G - Global Session Maintenance

Edit all session definitions that match user-specified search criteria. Sessions can be updated, added, deleted, and listed. Criteria can be saved for later use.

Note: For further information about the panels and fields used to perform these functions, see the online help.

Global Session Maintenance

Select option G - Global Session Maintenance from the MSDM primary menu or enter /MSDM.G at the ===> prompt to display the MSDM : Global Session Maintenance Menu.

The options available on the MSDM : Global Session Maintenance Menu are:

M - Maintain Search Criteria

Select this option to maintain search criteria stored when updating, adding, deleting, and listing session definitions.

U - Update Session Definitions

Select this option to update session definitions that match user specified search criteria.

A - Add Session Definitions

Select this option to add session definitions to users session lists that match specified search criteria.

D - Delete Session Definitions

Select this option to delete session definitions that match user specified search criteria.

L - List Session Definitions

Select this option to list session definitions that match user specified search criteria.

X - Exit

Select this option to exit this menu.

Example: Global Session Maintenance Update

This example shows the work flow process of an update to session definitions that match user specified criteria.

1. Select option U - Update Session Definitions from the MSDM : Global Session Maintenance Menu.

   The MSDM : Search Criteria panel displays.

2. Specify the Search Criteria.

3. Press F6=Action.

   The MSDM : Update Session Definition Details panel displays.

   The User ID field is mandatory and must be specified. You can specify a partial name suffixed by a * mask or * on its own to search through all session definitions. If you specify * on its own, you must specify at least one other criterion.

   Note: If you specify * in the User ID field, the time taken to return a response could be significant.

   All fields, excluding Jump Key fields and Yes/No fields, support * to match on all characters or ? to match on one.

   For example, if either of the following are specified in the logon request field:

   Logon Request ..... N??1 &?SER?D &USER* or

   Logon Request ..... N*

   The following logon request is matched:

   Logon Request ..... NM01 &USERID &USERPW; NM

   You can also store specified criteria and reload previously stored criteria.

   In this example, Forward Key is cleared from all session definitions that match the search criteria and MAI Menu Key 1 is set to F12.

4. Specify the required updates.

5. Press F6 (Action).

   The MSDM : Search Progress panel displays. This panel shows the number of sessions that match the search criteria and a graphical representation of this number.

   When the search is complete, the MSDM : Update Confirmation Panel displays. This panel shows the number of sessions that matched the criteria.

6. Press F6 (Confirm) to perform the search again.

   Repeating the search helps ensure that any sessions that have been updated while the confirmation panel is presented are included in the update.

7. When the search is completed, a panel showing the status of lock checking appears.

   Note: The lock checking process checks whether another user is updating any session definitions that match the specified criteria. If they are, the update process stops. You can rerun the process after the user has finished updating the session list.

   If no other users are updating the session lists that match the search criteria at this time, the MSDM : Maintenance Progress panel displays showing the status of the Global Session Update.

8. When the update is complete, the MSDM : Confirmation panel displays with a message stating whether the update was successful. If the update failed, review the activity log (/LOG) for additional messages that describe the reason for the failure.

9. If you have errors, enter **/LOG**.

Chapter 9: Setting Up Generic Resources

This section contains the following topics:

VTAM Generic Resources

CA SOLVE:Access supports the VTAM Generic Resource facility. This allows you to set up a group of CA SOLVE:Access regions to function as a single generic resource.

The VTAM Generic Resource facility allows a group of similar applications, such as a group of CA SOLVE:Access regions, to be associated with a generic name. An LU can be logged on to the generic name and then be passed to any of the participating applications, depending on load balancing.

VTAM keeps track of which application an LU is in session with. While the session is running, the LU and application are considered to have affinity. Any parallel sessions started by the LU to the generic group are passed to the same application.

Using a generic resource provides the following benefits:

- It permits session load balancing by distributing sessions between multiple applications.
- It increases application availability; if one application fails, failing sessions can be immediately reestablished with other applications in the group.

Generic Resource for CA SOLVE:Access Regions

CA SOLVE:Access regions require a full parallel sysplex to be available before they can participate in a CA SOLVE:Access VTAM generic resource group.

System Prerequisites for VTAM

To set up two or more CA SOLVE:Access regions to function as a generic resource group, you need a VTAM environment which meets the following requirements:

- The generic resource configuration is part of a parallel sysplex environment.
- A coupling facility resource management (CFRM) policy is active, and includes the structure ISTGENERIC.

Note: For more information about generic resources and how to define the ISTGENERIC structure, see the *SNA Network Implementation Guide*.

Each CA SOLVE:Access region in a generic group must satisfy the following requirements:

- The generic resource name must be specified in the GENRSRC JCL Parameter.
- A SOLVESSI started task must be running on each LPAR and each CA SOLVE:Access region running on each LPAR must be connected to the SOLVE Subsystem Interface (SSI).
- The same product configuration must apply to each CA SOLVE:Access region.
- Critical data sets must be shared; this sharing is best achieved by using Record Level Sharing (RLS).

  Data sets to be shared include:

  - SOLVE:Access Database (ACDB)—must be shared and enabled for RLS in a VTAM Generic Resource Environment
  - UAMS (if used)—must be shared and can optionally be enabled for RLS

System Prerequisites for VSAM RLS

Use of VSAM RLS requires the following software and hardware configuration:

- The systems must be configured as a parallel sysplex.
- The appropriate CF structures must be set up.

Note: For complete information about setting up systems for VSAM RLS support, see the *DFSMSdfp Storage Administration Reference*.

Data Set Requirements for Use With SOLVE VSAM RLS

For a VSAM data set to be eligible for SOLVE RLS, it must:

- Be SMS-managed
- Not be defined as recoverable

The Log (none) parameter is specified on the Define Cluster or Alter command. In addition, once a VSAM data set has been defined as using RLS, all concurrent users of the data set must open it using the RLS option.

How You Implement CA SOLVE:Access as a Generic Resource

To implement CA SOLVE:Access as a generic resource on a sysplex, complete the following tasks:

- Enable record level sharing (RLS) for the CA SOLVE:Access Database (ACDB).
- Set up the GENRSRC JCL parameter.
- (Optional) Enable RLS for your UAMS data set.

More information:

Enable Record Level Sharing (RLS) (see page 112)
Register to a VTAM Generic Resource (see page 113)

Enable Record Level Sharing (RLS)

Note: For complete information about configuring a parallel sysplex for VSAM RLS support, see the *DFSMSdfp Storage Administration Reference*.

All regions that are part of the CA SOLVE:Access VTAM generic resource group use the CA SOLVE:Access database.

To enable RLS

1. Ensure that VSAM RLS processing is available in your parallel sysplex.
2. Ensure that the Log(NONE) parameter was specified in the DEFINE CLUSTER or the ALTER CLUSTER command for the data set.

   A cache set must be specified on the storage class, and at least one of the associated CF cache structures must be available.
3. Ensure that the RLS VSAM Option is set in the $AC ACDB parameter group.
4. If this region was previously running with LSR, update to RLS and restart your region for the change to come into effect. You can also specify other options such as EXIT=NO.

Note: The ACDB is used for Session Definition information and Global Session Maintenance Search Criteria. In a VTAM generic resource environment, this file is also used for Disconnect and Single Sign-on user tracking.

If you specified to register this region to a VTAM generic resource during region setup, then the fields are already set to the correct value.

To enable UAMS for RLS

1. Ensure that VSAM RLS processing is available in your parallel sysplex.
2. Ensure that the data set is SMS managed and that the Log(NONE) parameter was specified in the DEFINE CLUSTER or ALTER CLUSTER command for the data set.
3. Specify a cache set on the storage class, and make sure that at least one of the associated CF cache structures is available.
4. Set the RLS VSAM Option for your UAMS data set by specifying JCL parameter XOPT=RLSU in your region's RUNSYSIN member.

Register to a VTAM Generic Resource

Registering CA SOLVE:Access to a VTAM generic resource makes the region a member of the VTAM generic resource group.

To register CA SOLVE:Access to a VTAM generic resource

1. Ensure that the ISTGENERIC Structure is available and active in your parallel sysplex.

   Note: For additional information about generic resource implementation, see the *SNA Network Implementation Guide*.

2. Ensure that the VTAM generic resource name is specified in the GENRSRC JCL parameter. Alternatively, specify the name in the ACINIT Customizer parameter group, which is accessed by entering **/PARMS** at the prompt.

   Note: If you update the ACINIT parameter group with a VTAM generic resource name, then file the change and it will take effect the next time the region is started. The region must register to the VTAM generic resource before the region's primary VTAM ACB is opened.

Note: The system parameter value shown in the Current Name field and the VTAM generic resource name can differ. You can update the VTAM generic resource name, file the change, and it is used at the next region restart.

Deregister from the VTAM Generic Resource Group

You can deregister a region from the VTAM generic resource group without having to restart the region.

To deregister your CA SOLVE:Access region

1. Enter **/PARMS** at the prompt to access the $ACINIT parameter group.
2. Enter **U** next to the group and specify NO in the VTAM Generic Resource Name field.
3. Press F6 to action the $ACINIT parameter group.

Note: If you want this region to register to the VTAM generic group when the region restarts, ensure that, after you deregister the region, you specify the VTAM generic resource name in the VTAM Generic Resource Name field and file the change. This action ensures that, when the region restarts, it is registered as a member of the CA SOLVE:Access VTAM generic resource group. If you leave this field as NO and file the change, the region does not register to the VTAM generic resource group when the region restarts.

Administration Utilities for Generic Resources

When CA SOLVE:Access regions are defined as belonging to a VTAM generic resource, users are attached to any of the CA SOLVE:Access regions that participate in the generic resource definition. Help desk staff need a way to determine which CA SOLVE:Access region a user is logged on to.

The exact domain name may be needed to provide the following functions:

- Broadcasting
- Manage active users
  - Monitoring user activity
  - Canceling a user
  - Disconnecting a user

Broadcasts to Generic Resources

This section describes how to use the Broadcast Services facilities of CA SOLVE:Access to enable broadcasts to members of the CA SOLVE:Access VTAM generic resource group.

These facilities are available from the Broadcast Services : Primary Menu (accessed by entering /BCAST at the command prompt).

The following broadcast options are available to all CA SOLVE:Access regions that form part of the VTAM generic resource group without performing additional configuration:

- Set and Store Generic Broadcast, accessed by entering **/BCAST.G** at the prompt
- Broadcast to all EASINET terminals and SOLVE users, accessed by entering **/BCASTSE.A** at the prompt

To enable user and terminal broadcasts to CA SOLVE:Access regions belonging to a VTAM generic resource group, there must be a broadcast system group corresponding to that generic resource.

Note: If the region is a generic resource, a system group is defined automatically to facilitate broadcasts to the local region only.

Enable Broadcasts to Generic Resources

To enable user and terminal broadcasts to regions belonging to a VTAM generic resource group, there must be a broadcast system group corresponding to that generic resource.

To add a broadcast system group corresponding to a generic resource

1. Enter **/BCAST** at the prompt.

   The Broadcast Services : Primary Menu appears.

2. Enter **LS** at the prompt.

   The Broadcast Services : Group List panel appears.

3. Press F4 (Add).

   The Broadcast Services : Group Definition panel appears.

4. Complete the following fields:

   Group Name

   Specifies the name of the new broadcast group.

   Description

   Describes the new broadcast group.

   Include Local System?

   Specifies whether the local system is included in the broadcast group. YES indicates that broadcasts can be issued on this system. NO indicates that broadcasts can be issued on remote systems only.

   Press F4 (Save)

   The group definition is saved. A message is returned, confirming that the new broadcast group has been added.

Add Resources to a Broadcast System Group

To add resources to a Broadcast System Group

1. From the Broadcast Services : Group Definition panel, press F5 (Resources).

   The Broadcast Services : Resource List panel appears.

2. Press F4 (Add).

   The Broadcast Services : Resource Definition panel appears.

3. Complete the following fields:

   Resource Type

   Specifies the type of resource to add. The following types of resource are supported:

   APPCLINK

   A predefined name of an APPC link between two regions.

   Limits: 12 characters

   DOMAIN

   A region (with a name of up to four characters) attached by INMC.

   Limits: 4 characters

   LU

   A network resource name that identifies the required region.

   Limits: 8 characters

   Note: When adding DOMAIN or LU resources, ensure that the necessary DEFLINK commands have been issued to allow the region to connect by using APPC.

   Resource Name

   Specifies the name of the resource to add. This name is used on the Broadcast Services : Send Menu when sending a broadcast to the system group.

   Press F3 (File).

   The changes are saved. The Broadcast Services : Resource List panel appears, with the new resource added.

4. Repeat steps 1 through 3 for each resource that you want to add to the broadcast group.

Send Broadcasts to Generic Resources

When you have defined a broadcast system group for your generic resource, you can use that system group to send broadcasts to regions belonging to the generic resource.

To send broadcasts to generic resources

1. Enter **/BCAST** at the prompt.

   The Broadcast Services : Primary Menu appears.

2. Enter **S** at the prompt.

   The Broadcast Services : Send Menu appears.

3. Complete the following field:

   System Group

   Specifies the system group name for your generic resource.

   Enter the mnemonic of the type of broadcast that you want to send at the prompt.

   The broadcast is sent.

Manage Active Users

This section describes the administration utilities available to assist the help desk staff or systems administrator to perform the following functions for CA SOLVE:Access regions that belong to a VTAM generic resource:

- Monitoring user activity
- Canceling a user
- Disconnecting a user

To list and manage active users

1. Enter the **/SS** panel shortcut.

   The Security and System Services : Primary Menu appears.

2. Type **LU** in the Select Option ===> field.

3. Type a value in the User ID field. This field allows you to enter the leading characters of a user ID. If you enter eight characters, then this value is used as an exact match. If you enter less than eight characters, then this value is treated as a prefix.

   If the last character is an *, it is ignored; that is, user IDs USER01 and USER01* are equivalent.

   The Security and System Services : Primary Menu displays.

4. If necessary, type a value in the Link or Domain Name field, which allows you to identify the CA SOLVE:Access region names (INMC link names) that apply to your request:

   - Leave blank for the local system.
   - Enter a specific link name.
   - Enter **?** to display a list of link names from which you can select one or more.
   - Enter ***** to denote all link names.

5. Press Enter.

   The System Support : User ID List panel displays. The System Support : User ID List allows you to identify which CA SOLVE:Access region (within the generic resource) each user is currently attached to. You can then apply any of the available actions to a particular user attached to a particular domain.

   Entries are displayed within their domains, with delimiter lines dividing the domains. The local domain (if selected) is always shown first, with others following in domain name (link name) order.

   Error messages, such as:

   ```
   NO MATCHING USER(S) ON THIS DOMAIN
   ```

   are displayed for any domain where they apply.

   Matching user IDs show the user ID and one of the following:

   - The terminal name (LU name) if logged on (signed-on user)
   - The disconnection data if applicable (disconnected user)

Actions on the User ID List

The following actions are available on the System Support : User ID List:

C (Cancel)

For a signed-on user, issues the CANCEL command specifying the LUNAME. For a disconnected user, issues the CANCEL command using the disconnection ID.

CF (ForceCancel)

For a signed-on user, issues the CANCEL command specifying the LUNAME and adds the FORCE option. Not valid for disconnected users.

D (Disconnect)

For a signed-on user, issues the DISCONN command. Not valid for disconnected users.

S (Show)

For a logged on or disconnected user, issues the SHOW MAI command for either the user (if disconnected) or the LU name (if signed on). For an error message line, provides online help for the message.

Each of these commands is issued using Command Entry, in the following ways:

- The SHOW MAI command is issued immediately.
- The CANCEL and DISCONN commands are preset into the Command Entry command line and you must action the command by pressing Enter. You can then issue any other SOLVE commands before using F3 (Exit) to return to the System Support : User ID List.

Chapter 10: MAI-FS Session Scripts

This section contains the following topics:

Session Scripts

The Session Scripting facility of MAI improves the presentation and ease of use of sessions.

Session scripts (referred to as script NCL procedures or, scripts) are NCL procedures that can interrupt a session to change or automate the processing that is performed. For instance, scripts can interrogate output from an application and automatically respond to it without the user being involved. Alternatively, scripts can interrogate input received from the terminal and change the data content, or perhaps alter the key that was used to enter the data. In this way, scripts can, for example:

- Complete the initial application logon procedures, automatically entering a user ID and password.
- Issue panels.
- Place the user into an initial transaction.
- Perform specific processing part-way through a session, such as switching from one transaction to another.
- Perform log off processing.
- Start scripts in other MAI sessions for the issuing user.

When a scripting procedure issues a panel, a panel request takes over the window and allows MAI to save the current screen image. When the procedure ends or issues an &PANELEND, MAI restores the screen image.

Examples of session scripts can be found in the CC16SAMP library. These scripts all start with the characters MAISCR.

Note: For information about NCL, see the *Network Control Language Programming Guide*.

How Session Scripts are Invoked

To invoke an NCL procedure as a session script for a particular session, enter its name in the appropriate field of the Logon Details panel. (Use of the Logon Details panel, which provides the various details and options required to start an MAI-FS session, is described in the *User Guide*). Enter the name of the script NCL procedure for the session, and also the parameters that are passed to the procedure as &1, &2, and so on. These logon details can be saved as stored definitions.

If a session has an associated script NCL procedure, the procedure can be run under the following circumstances:

- When the session is started.
- If the user enters a .S session skip command on an active session's screen (as long as the script procedure is not already running). Any parameters following the .S command are passed to the procedure as &1, &2, and so on. Enter these parameters after the .S command, in the same input field.
- By use of the SCRIPT START command.
- Under certain forced session termination conditions. In these cases, any parameters entered with the name of the procedure on the logon details panel are passed, exactly as at session start. The cases are:
  - The user logs off CA SOLVE:Access with MAI-FS sessions still active.
  - The user is canceled by an OCS operator.
  - The user's reconnect time limit expires.

The session with the user's terminal is lost and session reconnect is disabled on the system.

The MAI session inactivity time-out facility has invoked the script.

The script procedure uses the &MAISMODE system variable to decide which of the above three circumstances caused it to be started. If no processing is required the procedure ends. Once invoked, the script procedure is like any other NCL procedure, and most NCL facilities are available to it, such as file processing, message logging, command processing, and panel display.

The full set of system, global, and user variables is also available to the procedure. For instance, &LUNAME contains the name of the terminal from which the session was started, &LUROWS its screen size. &USERID and &USERPW can also be used when generating automatic replies to the application. MAI also introduces a number of new system variables that are of particular use in script procedures.

Once started, a script procedure runs for as long as required. The procedure can end at any time; in which case, session operation resumes normally.

Overview of Script Verbs and Variables

This section gives an overview of the various NCL verbs and system variables used by scripts. Those verbs and variables more commonly used are described first. The script can use the &MAISGET verb to obtain session definition details.

Note: For complete reference information for each system variable and verb, see the *Network Control Language Reference Guide*.

Note: The terms PLU and SLU are used. PLU stands for Primary Logical Unit, and is the application that the MAI session is logged on to. SLU stands for Secondary Logical Unit, and is the terminal being used.

The verbs and system variables are:

SCRIPT

Is a command that can be used to start and stop session scripts.

&MAISMODE

(Script Mode) Is a system variable whose contents enable the script procedure to determine how it was started. Scripts are started:

- At session start
- By the .S session skip command
- By a forced session end condition
- By the SCRIPT START command

&MAIOCMD

Is a system variable that returns the type of outbound data stream sent by the PLU.

&MAIREQ

Is a system variable that contains the contents of the Logon Request field of the MAI Logon Details panel.

&MAIREAD

Is a verb that suspends execution of the procedure, pending arrival of the next data stream from either the PLU or the SLU (application or terminal) or for specified time interval to elapse.

The &MAIREAD statement nominates the data in which it is interested as PLU, SLU, or ANY (PLU or SLU). Data that is not nominated as required is processed normally, being sent on to the session partner.

When the required data stream arrives, the procedure resumes execution, and then decides on a course of action.

This action can involve deleting the data stream (not sending it on to the session partner), changing it, replacing it with a different data stream or sending it on intact.

The verb can also specify a time interval after which the NCL procedure regains control even if the requested data stream has not arrived.

&MAIFRLU

(From LU) Is a system variable that shows the direction of the data flow after an &MAIREAD, as either PLU or SLU. In this way, the procedure can determine whether the PLU or SLU sent the data.

&MAIFIND

Is a verb that enables the procedure to determine whether the data received contains a specific string. The required string can be specified in either character or hexadecimal notation.

&MAIPUT

Is a verb that enables the procedure to set up a data stream to be sent to the application (PLU). &MAIPUT fills in the input fields on the panel that is the current logical screen image. Multiple &MAIPUT statements can be issued to build up the data stream. Each one supplies the contents of one input field. &MAIPUT statements can be used to reply to PLU data, or to change the input from the terminal (SLU) before sending it on to the application.

Issue an &MAIINKEY statement after the &MAIPUT statement to base the data stream length on the key type.

&MAICROWS

Returns to a script the current number of rows on the MAI session's virtual terminal.

&MAICCOLS

Returns to a script the current number of columns on the MAI session's virtual terminal.

&MAICURSA

(Cursor Address) Is a verb that supplies cursor positioning information. The verb is used with &MAIPUT when sending data to the PLU.

&MAICONT

(Continue) Is a verb that determines whether to send the last received data (or data built up using &MAIPUT) to the PLU, the SLU, or BOTH.

Data is sent to the PLU under any of the following conditions:

- Data has been received from the SLU and is to be forwarded to the PLU.
- Data has been received from the PLU and a reply built up using &MAIPUT is to be sent back to the PLU.

Data is sent to the SLU when data has been received from the PLU and is to be forwarded to the SLU.

Data is sent to BOTH under the following condition:

- Data has been received from the PLU.
- Data is to be forwarded to the SLU.
- A reply built up using &MAIPUT is to be sent back to the PLU.

A further option on &MAICONT controls whether data sent on to the SLU, the physical terminal, must be displayed, or whether the display is optional. If it must be displayed, the procedure is suspended until the display has been accomplished, which could mean waiting until the terminal user jumps back to display the session's screen image.

Optional display is often used when the procedure is replying to the application and wants the user to see the application's output, but only if the user is displaying that session. Otherwise, the procedure is free to continue its dialog with the application.

&MAIDEL

(Delete) Is a verb that deletes the data stream last received, that is, the data stream is not sent to the terminal or the application.

&MAIINKEY

(Input Key) Is both a verb and a system variable. As a verb, it is used to supply the value of the key to press when sending data to the application (for example, Enter, F3, or PA2). Issue an &MAIPUT after &MAIINKEY to extend the data stream.

As a system variable, &MAIINKEY makes available the key that the user pressed to enter data at the terminal.

&MAIUNLCK

(Unlock) Is a system variable that indicates whether the data stream received from the PLU would unlock the keyboard if sent on to the terminal.

Session scripts receive each data stream as the PLU sends the stream. Sometimes, it is necessary to wait for a subsequent data stream to unlock the keyboard before an automatic reply can be generated.

&MAISID

(Session ID) Is a system variable that contains the ID of the session for which the script procedure is running.

&MAIAPPL

(Application) Is a system variable that contains the VTAM name of the application connected to the session.

This name can vary during the session if an application issues a VTAM CLSDST/PASS to pass the session to another application.

&MAILU

(Logical Unit) Is a system variable that contains the name of the VTAM ACB used by MAI to represent the secondary end of the session.

&MAIDSFMT

(Data Stream Format) Is a verb that makes the current data stream available to the procedure. The data stream is placed in character format hexadecimal in variables nominated as parameters for the statement.

&MAIREPL

(Replace) Is a verb that allows the procedure to replace the current data stream destined for the terminal. The replacement data stream is supplied in character format hexadecimal.

&MAIWNDOW

Is a system variable that indicates the visibility of the related MAI-FS session. A value of FOREGROUND is returned when the application is currently displayed. BACKGROUND indicates that the session is not displayed.

Data Stream Received by &MAIREAD

An &MAIREAD statement is satisfied by the arrival of a data stream from either the PLU or the SLU, as nominated on the statement itself. *Each data stream sent by the PLU or SLU can satisfy an &MAIREAD.*

Some applications build up a screen image with multiple data streams. For instance, when logging on to TSO, multiple I/O operations can take place, perhaps sending various broadcast messages and finally the READY prompt. Each of these can satisfy &MAIREAD. However, only when a data stream capable of unlocking the keyboard is received should an automatic reply be generated.

In SNA terms, each data stream delivered to &MAIREAD is a complete chain. If MAI receives a data stream in multiple elements, they are pieced together before being delivered.

Application Output Display

A VIEW option is available on the &MAICONT statement to control the way in which application output is presented to the terminal user. The procedure has the following options:

- Not displaying application output on the terminal
- Displaying output only if the user is actually viewing the session at the time
- Waiting until the user has viewed the output

Some data streams should be sent to the terminal. An example of this situation is a Read Partition Query command sent to the terminal by some applications to retrieve terminal characteristics. The terminal hardware itself replies to this command. The procedure must be aware when this type of data stream is received. Use the VIEW=WAIT option on &MAICONT to help ensure that the data stream gets sent to the terminal and ensure that the reply gets sent back to the application. For example:

```
&MAIREAD PLU * wait for query
&MAICONT SLU VIEW=WAIT * send it to the terminal
&MAIREAD SLU * wait for terminal's reply &MAICONT PLU * send it to the application
```

Note: You can simulate the hardware response to a Read Partition Query. If this simulation is required, use &MAIINKEY SF to indicate that the script is building an inbound structured field. Then use &MAIPUT (without the ROW or COL operands) to build the required response. Then use &MAIPUT PLU to send the data stream to the application.

Terminal Input Control

Script procedures have full control over when terminal input is allowed. The &MAIREAD statement can nominate to deliver SLU input to the procedure (by specifying SLU or ANY). From there, the procedure can decide whether the input is sent on to the PLU, or discarded by issuing &MAIDEL.

Alternatively, the procedure can request that only PLU data be delivered to it by using &MAIREAD PLU. In this case, terminal input is sent on to the PLU automatically, bypassing the procedure entirely, unless the terminal did not receive the last data stream sent from the PLU. This scenario is the case where a script procedure is concerned with generating automatic replies to the application, for instance, to perform automatic logons. The procedure issues an &MAIREAD PLU to receive data streams from the application, so that those data streams can be searched using &MAIFIND and replied to using &MAIPUT and &MAICONT. If the procedure replies without letting the terminal user see the session output, terminal input is automatically blocked.

Effect of Session End

If an MAI session ends while a script is still running, the script is automatically flushed by MAI. This applies for both normal logoff and the use of the cancel commands from the MAI menu and as session skip commands.

If a script is running when a forced session end condition occurs, the script is flushed before being rerun for forced session end processing.

Session End Script

The script procedure allocated to a session is started under a variety of forced session end conditions, such as the user logging off with MAI sessions still intact. The procedure can attempt to end the session normally if required, by using &MAIPUT and &MAICONT PLU to issue logoff type commands and waiting for the appropriate responses.

The session end script runs in the BSYS environment. This allows the script to execute even though the original user's region has been terminated. The &USERID system variable contains the BSYS user ID and &ZOUSERID returns the user ID of the original user.

If the script ends before the session itself ends, the session is canceled by MAI, as is the case if there is no script procedure allocated to the session. Therefore, if a script decides not to instigate automatic logoff processing it should end to allow the cancel to take place. Conversely, if logoff processing is instigated the procedure must ensure that it does not end before the session ends. This is normally accomplished by issuing an &MAIREAD after the last message from the application is received. The procedure will wait on this statement until it is flushed by the session ending.

Session Skip Script

The .S session skip command entered by the user starts this script procedure, unless a script is already running. The procedure is free to perform any processing required, which will normally involve sending data to the application using &MAIPUT and &MAICONT PLU. The procedure may execute for any length of time. When it ends, the session resumes normal processing.

Script Diagnosis

You can diagnose problems with scripts in many ways, for example:

- The &CONTROL TRACE or &CONTROL TRACELOG statements to trace execution flow. With the TRACE options in effect, MAI additionally traces all data streams sent to the session partners. Specifying &CONTROL TRACELOG results in execution flow and all data streams being traced to the SOLVE activity log.
- &WRITE statements at appropriate places in the procedure can also aid problem diagnosis.
- Use of the SOLVE DEBUG facility.

  Note: For information about using DEBUG, see the *Network Control Language Programming Guide*.

The SHOWMSG primary command (can be abbreviated to SHOW) provides access to any line messages produced by scripts executing on behalf of the user. The messages are displayed on one or more full-screen panels. If more than one script has produced messages, they are intermixed. Press Enter to display the next panel.

The SOLVE region parameter NMIQLIM limits the maximum number of queued messages.

Errors resulting from the execution of the script are written to the activity log. If a script is flushed because of an error, the session continues as though the script had ended normally, except for a session end script in which the session is terminated.

Chapter 11: Generating Logon Scripts

This section contains the following topics:

About Logon Scripts

An alternative to writing an MAI script manually is to use the Logon Script generation function. This works as a record and playback type process. The generated member is stored in the region's TESTEXEC library.

Generate Logon Script from the Administration Menu

The Administration menu (/ACADMIN) supports option G - Generate a Logon Script. This function can also be accessed using the /ACSGEN shortcut.

To start script generation from the Administration menu, select Option **G**.

A panel to select the application for which to generate the script appears. This panel lists all the DEFLOGON definitions defined by the $ACINIT procedure used in the $AC ACINIT Customizer parameter group at startup.

Generate Logon Script at Session Start

You can also start the script generation process by activating a session where the script defined for the session is '$ACSCRPT'.

To start script generation using $ACSCRPT

1. Enter **L** next to the session on the MAI Primary Menu.

   The Session panel appears.

2. Specify $ACSCRPT in the Script Name field, and press Enter.

   The logon recording and script generation process starts.

Generate a Logon Script

To generate the script:

1. Enter initial details including a function key that signifies recording end, and press F6 (Action).

   The application session starts. The next screen displays from the application and is most likely a welcome or logon screen that includes user ID and password fields. If the session fails to start (for example, the application is inactive), then the user is returned to the initial details panel with an error message displayed.

2. Enter information for the recording process fields, as required, and press the appropriate keys for the application.

   Note: Your user ID and password are generated as variables (&USERID, &USERPW), so the script is not specific to the user performing the recording.

3. When the application is displaying the appropriate panel, such as the primary menu, press the function key delegated to terminate recording.

   The Save Script panel displays.

4. Enter information for the Save Script process panel, which contains information about where the generated script is stored. The default value is the TESTEXEC library for the region. This data set is a library in the COMMANDS DD concatenation so that this region can use the script.

   Press F6 (Generate).

   When the script is generated successfully, the Confirm Procedure script review panel appears.

5. Review and confirm the information for the script. The generated script displays in browse mode and you can scroll to review the processing generated.

   Press F6 (Confirm).

   The script is written to the delegated output PDS member.

   After the script has been saved, you are returned to the application session, where you must then log off and terminate the application session.

Initial Details Panel

When the script generation process is initiated, the Logon Recording : Recording Detail panel displays.

Save Script Panel

The save script panel displays when the recording termination key is pressed. It prompts for information about where the generated script will be saved. If you are recording a new version of a script, you may want to save the existing version before overwriting it. By default, the generation process does not overwrite an existing member.

Generated Script Details

The generated script has the following characteristics:

- A comment header containing information about the script generation.
- A section supporting the logon process that is labeled.MODESTART. The code following this label is generated based on the recording process. You can modify the generated logon process to handle different screen sizes.
- A section supporting session end processing that is labeled.MODEEND. This section of code is executed when a user is canceled and performs an application logoff.

Example: A Generated Script

```
-*************************************************************************
-*                                                                       *
-* NAME : LGNSOLVE                                                       *
-*                                                                       *
-* TITLE : LOGON SCRIPT                                                  *
-*                                                                       *
-* PURPOSE : AUTOMATE SESSION ESTABLISHMENT                              *
-*                                                                       *
-* SYNOPSIS : SOLVEPROD.TESTEXEC (______)(LGNSOLVE)                      *
-*                                                                       *
-* CREATE : MON 30-JUN-2003 20.04.06                                     *
-*                                                                       *
-* USERID : USER01                                                       *
-*                                                                       *
-* DESCRIPTION : THIS SCRIPTING PROCEDURE AUTOMATES THE SIGNON TO        *
-* SOLVE:Access                                                          *
-*                                                                       *
-*************************************************************************
&CONTROL -* RESET ALL TO DEFAULT
&CONTROL NOCMDSEP NODUPCHK NOENDMSG NOINTLOG NOLABEL NORECCHK +
         NOUCASE NOVARSEG FLDCTL PFKALL MDOCHK
```

```
-*
-* Select processing based on &MAISMODE
-*

&GOTO .MODE&MAISMODE
&RETURN
-*************************************************************************
-*                                                                       *
-* SUBROUTINE : MODESTART                                                *
-*                                                                       *
-* SCRIPT MODE : START                                                   *
-*                                                                       *
-* PURPOSE : AUTOMATE APPLICATION LOGON AND RESOURCE DISPLAY             *
-*                                                                       *
-* DESCRIPTION : This section is invoked to handle the initial           *
-* dialogue with the external application. It logs on                    *
-* the user and if a parameter is available issues                       *
-* a command to display a resource                                       *
-*************************************************************************
.MODESTART

&CALL PROC=$ACSCALL SHARE=($AC>) +
      PARMS=(ACTION=CONTROL,VIEW=ALL)
&CALL PROC=$ACSCALL SHARE=($AC>) +
      PARMS=(ACTION=QUERYWT1)

&CALL PROC=$ACSCALL SHARE=($AC>) +
      PARMS=(ACTION=SETFLD,ROW=21,COL=40,+
      DATA='&USERID')

&CALL PROC=$ACSCALL SHARE=($AC>) +
      PARMS=(ACTION=SETFLD,ROW=21,COL=64,+
      DATA='&USERPW')
&CALL PROC=$ACSCALL SHARE=($AC>) +
      PARMS=(ACTION=SETCSR,ROW=21,COL=71)
&CALL PROC=$ACSCALL SHARE=($AC>) +
      PARMS=(ACTION=KEY,KEY=ENTER)

.ENDEND
&RETURN
-*************************************************************************
-*                                                                       *
-* SUBROUTINE : MODEEND                                                  *
-*                                                                       *
-* SCRIPT MODE : END                                                     *
-*                                                                       *
-* PURPOSE : TERMINATE SESSION                                           *
-*                                                                       *
-* DESCRIPTION : THIS SECTION IS INVOKED IF A USER IS CANCELLED AND      *
-* THE EXTERNAL APPLICATION SESSION IS STILL ACTIVE.                     *
-* THE SCRIPT SHOULD LOGOFF THE USER.                                    *
```

```
-*                                                                      *
-************************************************************************

.MODEEND
   -* This code is driven when a user is cancelled.
   -* Add code here to logoff from the application.
   -* By ensuring a clean logoff there is no chance of the session
   -* on the virtual terminal being connected to a subsequent
   -* user of that same virtual terminal.
&RETURN
   -* End of Script
```

How You Test the Generated Script

After you generate the script, test it before making it available to your users.

Note: If you have used the script previously, then delete the old copy from storage by issuing SYSPARMS UNLOAD=*name*.

On the MAI menu, issue an **L** command against a session definition that uses the application for which you have recorded the logon sequence. The session details panel displays. Enter the generated script member name in the Script Name field, and press Enter. The session starts using the script.

Note: Some applications behave differently on different screen sizes. For example, Model 2 (24x80) can have a different screen layout to a Model 4 (43x80). In this case, to have a script work for all screen sizes, you can modify the script to include conditional logic. See the following example code fragment:

```
&IF &MAICROWS EQ 24 &THEN +
  &CALL PROC=$ACSCALL SHARE=($AC>) +
    PARMS=(ACTION=SETFLD,ROW=21,COL=40,+
    DATA='&USERID')

&ELSE +
 &IF &MAICROWS EQ 32 &THEN +
  &CALL PROC=$ACSCALL SHARE=($AC>) +
    PARMS=(ACTION=SETFLD,ROW=29,COL=40,
    +DATA='&USERID')
&ELSE +
&IF &MAICROWS EQ 43 &THEN +
  &CALL PROC=$ACSCALL SHARE=($AC>) +
    PARMS=(ACTION=SETFLD,ROW=40,COL=40,+
    DATA='&USERID')
```

How You Implement the Generated Script

After you have tested the script, you can make it available to users generally by any of the following methods.

- Use MSDM to update individual session definitions
- Use the MSDM Global Session Maintenance function to update multiple definitions.
- Update the appropriate DEFLOGON statements in your installation initialization member to include SCRIPT=*name*.

Note: Changes to session definitions take effect when the user next logs on.

$ACSCALL Programming Interface

$ACSCALL is an API used by the generated script to control the session. This API supports arguments in the form *keyword*=*value*. The ACTION keyword supports the following values: CONTROL, SETFLD, SETCSR, KEY, QUERYWT1, QUERYWT2, PAUSE, and WAITFOR.

ACTION=CONTROL

```
$ACSCALL ACTION=CONTROL [ VIEW=[ALL | NONE] ] [ TRACE=n ]
```

CONTROL

Used to modify the performance or behavior of the scripting process.

VIEW

Controls whether outbound data streams are sent to the terminal. Use VIEW=NONE if you do not want the user to see the panels that the script has responded to (a flickering effect as the screen changes quickly). However, use of this option for some applications can result in an incomplete screen display. If so, then either use VIEW=ALL or modify the script so that it sends a screen refresh request to the application before terminating.

TRACE

Used to set a tracing level for debugging the script. The value is a number in the range 0 through 2.

- 0—No tracing.
- 1—Log message after each call.
- 2—NCL trace log each call (this results in a dump of data streams sent and received).

ACTION=SETFLD

```
$ACSCALL ACTION=SETFLD [ ROW=nn ] [ COL=nn ] [ DATA=input_data ]
```

The SETFLD action specifies the data for input fields. The script generation process generates a call for each field entered by the user and also for each field automatically returned by the terminal. (Some applications send output fields with the 'modified data' tag set so that the associated data is returned.).

If your application changes the field positions when using different screen sizes, then you may need to modify generated scripts to include conditional logic to handle this situation.

ACTION=SETCSR

```
$ACSCALL ACTION=SETCSR ROW=nn COL=nn
```

The SETCSR action specifies the screen location for the cursor.

ACTION=KEY

```
$ACSCALL ACTION=KEY KEY=keyname
```

The KEY action results in the input data stream being sent to the application. Valid values are F1 to F24, PA1, PA2, PA3, and CLEAR. For function keys, all data from SETFLD requests and SETCSR is included. For program attention keys, the SETCSR data is included.

ACTION=QUERYWT1

```
$ACSCALL ACTION=QUERYWT1
```

Terminals that support extended attributes are queried to determine their capabilities. The QUERYWT1 action is generated to handle this process by script recording. The action results in the query request being forwarded to the terminal and the terminal's response being sent to the application.

ACTION=QUERYWT2

```
$ACSCALL ACTION=QUERYWT2
```

The QUERYWT2 generates an automatic response to a terminal query without involving the physical terminal. An RPQ response is sent that indicates support for standard color and highlighting. The recording process does not generate the QUERYWT2 action.

You can use this action to avoid physical terminal interaction during the logon process but still have color and highlighting available.

ACTION=PAUSE

```
$ACSCALL ACTION=PAUSE MAXWAIT=nn COUNT=n
```

This function is used to handle multiple consecutive outputs from the application and is generated by script recording. For example, after initial logon to an application, there can be numerous initial messages such as broadcasts. These messages are extraneous to the conversation and are ignored. The MAXWAIT is the time in seconds to wait for additional output. If the time limit expires it is assumed that the output sequence is complete.

ACTION=WAITFOR

```
$ACSCALL ACTION=WAITFOR DATA='string'
```

This function is used to handle multiple consecutive outputs from the application where a particular string indicates that the output sequence is complete.

Chapter 12: Advanced MAI-FS Customizing

This section contains the following topics:

ACB Sharing Considerations

MAI-FS is capable of conducting multiple sessions on behalf of a user from a single VTAM ACB (LU name). For example, a single ACB can be used to create sessions with IMS, CICS and TSO simultaneously. This is known as ACB sharing and is desirable because it reduces the number of VTAM APPL statements required to support a given number of users, with a corresponding reduction in VTAM overheads.

Note: ACB sharing is not available when the MAI service subtask is not used.

Pool ACBs

If MAI-FS requires an LU name from a pool, it checks whether an LU name that is already in use for this user can be shared. Sharing is possible if the following criteria are satisfied:

- The DEFLOGON table entry that represents the application with which the session is to start has the MSHR=YES specification.
- An LU name can be found that is in use and was itself opened shareable and that does not already have a session with the desired application.

If a shareable LU name cannot be found, a new ACB is opened and set shareable by future sessions if the DEFLOGON table entry specifies MSHR=YES.

Specific ACBs

If MAI-FS requires a specific LU name, it can share this LU name for multiple sessions only if the MAIEX02 installation exit is used and signifies sharing.

In this case, MAI-FS shares the LU name if the following criteria are satisfied:

- The specified LU name is already in use and was itself opened shareable (not from a pool).
- The LU name was not originally allocated from a pool.
- The LU name does not already have a session with the desired application.

If a shareable LU name cannot be found, the specified ACB is opened and marked shareable by future sessions if MAIEX02 authorized it.

ACB Sharing Restrictions

Certain restrictions apply to the use of ACB sharing. While it can be used easily with most applications, certain considerations apply when using ACB sharing with applications that use the VTAM CLSDST/PASS function to pass terminals from one application to another. The rules that apply are summarized as follow:

- The APPL= operand on the DEFLOGON command must always refer to the network name of the target application. This is because the network name is the name made available to the various MAI VTAM exits and they must be able to correlate session information.
- ACB sharing can always be used to create MAI sessions with applications that do not CLSDST/PASS their terminals to other applications.
- The new (passed to) application name must be a unique match with the original (passed from) application name for the number of characters specified by the MAIACBLN system parameter before MAI will accept the session. If MAIACBLN is set to 0, the original application name must be a subset of the new application name. The following examples illustrate the third criterion. In these examples, the original application (APPL1) name is TSOA.

```
MAIACBLN=0 APPL2=TSOA**** the session will be accepted.
MAIACBLN=1 APPL2=T******* the session will be accepted.
MAIACBLN=1 APPL2=?******* the session will be rejected.
MAIACBLN=2 APPL2=TS****** the session will be accepted.
...
MAIACBLN=4APPL2=TS0A****the session will be accepted.
```

The asterisk(*) character represents any single alphanumeric character and a question mark(?) represents any single alphanumeric character except T.

MAI-FS Sessions to Target Applications

An MAI-FS session with a target application is started when a user selects one of the logon options from the MAI : Primary Menu.

A logon request forms part of a logon option. This request is validated against the DEFLOGON entry table to provide the name of the application with which the session is to start.

MAI-FS attempts to open a session with the selected application. MAI-FS emulates an LU Type 0 device (3277) or an LU Type 2 device (3278/3279), acting as the secondary end of the session. The application sees the MAI-FS connection as a standard session with a physical 3270-type terminal and is unaware that the session is really with another application.

Certain application subsystems, such as IMS, require that every logical unit with which they are to have a session, be defined to them before any session with the LU is allowed. Other systems, such as TSO, have no such restrictions and accept a session from any supported device type that issues a logon request.

If MAI-FS is to establish sessions with systems such as CICS or IMS, the VTAM LU names that MAI-FS uses must be included in the appropriate system generations, together with any other relevant information, for example:

- Ability of the logical unit to act as a master terminal
- Its authority level
- Other details that depend upon the requirements of the system to which the logical unit is being defined.

The definition to a subsystem such as CICS or IMS is the same as for a physical 3270-type device, either LU Type 0 or LU Type 2.

Cross-Domain MAI-FS Sessions

Unless VTAM has been configured to support dynamic cross domain definition and adjacent SSCP lookup, two conditions must be satisfied for a CA SOLVE:Access user in one domain to successfully request an MAI-FS session with a target application running in another domain:

- The appropriate cross domain definitions must have been filed in the VTAM definition library and activated.
- The target subsystem must have the MAI-FS LU name defined to it as a valid terminal if required.

Multiple Region Considerations

The discussion of MAI-FS so far has assumed one SOLVE region with MAI-FS sessions to one or more target applications. When more than one SOLVE region in a network has MAI-FS sessions to the same set of target applications, each SOLVE region should use a unique set of ACB names. These names were defined when you installed and set up your product and specified in the EXTAPPLPOOLS parameter group in Customizer.

Note: For more information, see the *Installation Guide*.

This implementation allows the names used by each SOLVE region to be associated with the domain in which that CA SOLVE:Access is executing. It avoids VTAM definition conflicts when attempting to start cross-domain sessions.

The other advantage of assigning each SOLVE region its own set of MAI-FS LU names is that it allows added control over which SOLVE regions can establish MAI-FS sessions with which subsystems. You include definitions in the subsystem generation only for terminals used by selected SOLVE regions within the network.

More information:

MAI-FS Operational Scenario (see page 233)

Logmode Tables

MAI sessions are established between the target application, which is the primary end of the session and the MAI application that is associated with the LU name selected when the session is established.

Although the MAI end of the session is really a VTAM application, it emulates precisely the protocols of an LU-Type 1 3767 device, an LU-Type 0 3277 device, or an LU-Type 2 3278/9 device. The implication of this is that the BIND parameters associated with the session must be appropriate to the management of sessions of the appropriate type.

When an MAI session is started, it is logically equivalent to a logon operation from a real terminal. As with a real logon operation, the BIND parameters associated with the session are those defined in the VTAM logmode table entry that is associated explicitly or by default with the secondary LU of the session, in this case the APPL definition being used by MAI to establish the session.

To ensure that the correct BIND parameters are associated with the MAI APPL definitions, the APPL definition statements should include a MODETAB operand identifying the name of a logmode table that is to be associated with the APPL definition.

Class of Service parameters may be added to BIND parameter definitions if MAI sessions are to travel on particular virtual routes, or are to have a particular priority.

Logmode Entry Selection with Standard Terminal Devices

The sample MAI-FS APPL definitions included in the MSSAMP distribution library include a MODETAB statement that refers to the logmode table MAIFMODE. This is the recommended way of coding these definitions if MAI is to be used only from standard terminal devices. In this context, standard terminal devices are terminals that support standard IBM BIND interpretations only.

MAI determines the physical characteristics of the terminal from which it is being used and generates the name of a logmode entry that adequately describes the terminal. This name will match the name of an entry in the MAIFMODE logmode table. For example, if the terminal is a non-SNA Model 2 terminal, MAI-FS will generate the name M2NSNA. This name corresponds to the name of a logmode table entry in the MAIFMODE table containing BIND parameters appropriate for non-SNA Model 2 terminals. If the terminal is an SNA Model 3B (supporting extended data streams), MAI-FS will generate the name M3SNAQ (where Q means the device supports the read partition query command).

Logmode Entry Selection with Nonstandard Terminal Devices

Terminal devices using nonstandard BIND interpretations (for example, Fujitsu terminals) cannot be used to their full capabilities if an entry is selected from the MAIFMODE table supplied. The MAIMDTAB system parameter can be used to specify the name of the logmode table containing the entries to use for sessions between MAI-FS and any application. If this parameter is not set, MAI-FS uses the MAIFMODE table.

Logmode table entries matching the characteristics of all terminal types from which MAI can be used should be included in the table specified by MAIMDTAB.

When the terminal is initially logged on, the table is searched for an entry matching the BIND parameters used for the session between the terminal and CA SOLVE:Access. If a matching BIND is found, the logmode entry name is retained for use by MAI on sessions between MAI and any application entered into on behalf of the terminal.

If no matching BIND is found, MAI-FS generates a logmode entry name using the process for standard terminal devices. If there is no entry in the logmode table with a name matching that generated by MAI-FS, the session request fails. For this reason, it is recommended that the logmode entries supplied in the MAIFMODE table are copied into the logmode table specified by the MAIMDTAB system parameter.

For MAI-FS to use this logmode selection process successfully, the MODETAB statement of the MAI APPL definitions must refer to the logmode table specified by the MAIMDTAB parameter. The logmode table must be assembled and linked into a load library accessible to CA SOLVE:Access, and the appropriate VTAM library.

Selection of a Specific Logmode Entry

The provision of a particular LOGMODE name on the MAI-FS Logon Details panel overrides the name chosen by MAI-FS. In this case, the name entered must be either one of the supplied names or the name of an entry added to the table. Entry of a name in this manner would normally only be necessary to specify, for example, a particular screen size, or particular class of service, or when certain applications require their secondary LUs (in this case the MAI-FS applications) to have a particular logmode table entry name.

MAI-FS Use of Session Protocols

As described in the previous section, MAI-FS presents a set of session parameters when a session is requested that adequately describe the physical terminal, for example:

- If the terminal is a 3277, the session parameters describe the LU0 protocols.
- If the terminal is a 3278 attached to an SNA 3274 control unit, the session parameters describe the LU2 protocols.

Applications that do not override the session parameters presented to them at logon time then operate the session as described by the session parameters. The session appears to the MAI-FS user exactly as it would look if the applications were not operating through MAI-FS. As an example, the RESET key on a 3277 can be used to unlock the keyboard and data can be typed ahead, as is the case when using PA1 to interrupt a process on a TSO session.

This facility is not available on an LU2 terminal because after the keyboard is locked it cannot be unlocked using the RESET key. Terminals that use this session type have an ATTN key to signal the application. These facilities can be used when operating through MAI-FS if the session parameters match the attributes of the physical terminal.

A problem arises when the session parameters used on a session do not match those parameters of the physical terminal. Consider the following case:

- A session is started to CICS from a 3277 LU0 terminal.
- MAI-FS chooses LU name NMMAF003, presenting session parameters associated with the M2NSNA table entry to describe the physical terminal.
- The NMMAF003 LU is described to CICS as an LU2 terminal and so it modifies the session parameters to reflect that device type.
- These modified parameters are presented to MAI-FS when the session is created.

Under these circumstances, MAI-FS operates the session to the target application according to the session parameters as modified by the application (LU2). However, data streams received from the application are sent to the physical terminal according to that terminal's requirements (LU0).

The only problem that arises here is the use of the RESET key on the 3277 terminal. If the keyboard is locked, and the RESET key is pressed and data typed ahead, MAI-FS ignores that data. The session protocols used on the MAI session do not allow this operation to happen. However, this problem is not normally present, as the types of application to which terminals have to be defined (IMS and CICS) would not typically use this mode of operation.

Obviously, the reverse could apply where the physical terminal is an LU2 3278, but the definition is of an LU0. In this case, the problem that arises is the use of the ATTN key on the terminal. Because the MAI session is LU0, there is no means of conveying the fact that the ATTN key has been pressed to the application. Once again, this problem is not normally present, as the types of application to which terminals have to be defined (IMS and CICS) would not typically use the ATTN key.

The ATTN key is handled in a special way by MAI-FS. When pressed, it causes the keyboard to be unlocked to enable a jump away from the session. If pressed a second time, a signal is sent to the application, if the session type is LU2.

Note: For more information, see the *User Guide*.

Summing up, it is adequate to define MAI-FS LUs to applications such as IMS or CICS as LU2 nodes and sessions would operate satisfactorily from any physical terminal. However, the explanation has been given to assist in catering for special circumstances.

MAI-FS Use of Screen Sizes

As described previously, MAI-FS presents a set of session parameters that adequately describe the physical terminal when a session is requested and those session parameters include the size of the physical terminal. For example, a model 2 has one size only, 80 columns by 24 lines, but a model 4 has two sizes, 80 columns by 24 lines and 80 columns by 43 lines.

Applications that do not override the session parameters presented to them at logon time then operate the session as described by the session parameters. They can use the size of the terminal as they see fit.

As with session protocols, a problem arises when the session parameters used on a session do not match those parameters of the physical terminal. Consider the following case:

- A session is started to IMS from a 3278 model 5 (27 line by 132 column) LU Type 2 terminal.
- MAI-FS chooses LU name NMMAF012, presenting session parameters associated with the M5SNA logmode table entry to describe the physical terminal.
- The NMMAF012 LU is described to IMS as a model 2 terminal and so it modifies the session parameters to reflect that screen size.
- These modified parameters are presented to MAI-FS when the session is created.

Under these circumstances, IMS always sends model 2 data streams to MAI-FS. MAI-FS controls the physical terminal accordingly and switch the model 5 terminal to its default screen size (model 2) whenever a data stream is sent to it from the IMS session.

Now consider the following situation:

- A session is started to IMS from a 3278 model 2 LU-Type 2 terminal.
- MAI-FS chooses LU name NMMAF012, presenting session parameters associated with the M2SNA logmode table entry to describe the physical terminal.
- The NMMAF012 LU is described to IMS as a model 3 terminal and so it modifies the session parameters to reflect that screen size.
- These modified parameters are presented to MAI-FS when the session is created.

Under these circumstances, IMS would send model 3 data streams, but these data streams could not successfully be sent to the physical device because it is only a model 2. For this reason, MAI-FS refuses the session, recognizing that the physical terminal does not support the screen sizes presented in the session parameters.

If the installation wants to create MAI-FS sessions (with applications that require predefined LUs) from non-model 2 terminals and the session is to operate in non-model 2 mode, a specific LU name may be required when the session is created (using the NODE NAME field on the logon details panel). This node name is then defined to the application as being of the same screen size as the physical terminal.

Alternatively, you can use the MAI user exit MAIEX02. The exit can supply a node name automatically or a specific node name prefix to allow MAI to choose a node name from a pool. All node names in this alternative pool could then be defined to the application as the appropriate size terminal. This technique is made possible by the fact that the exit is supplied with the size of the physical terminal and can decide which pool to choose.

Application-Specific Online Help

Online help is available about all aspects of MAI-FS operation. Online help is obtained by entering the HELP primary command or pressing the F1 or F13 key at the MAI : Primary Menu, or by entering the question mark (?) session command from an MAI-FS session. MAI online help is defined and displayed using the Common Application Services (CAS) Help Manager.

A request for session online help information drives an NCL procedure named $MAIHELP, which is supplied with MAI. The supplied procedure displays session details and provides entry into the CAS Help Manager to display the MAI online help. The procedure also enables application-specific online help to be accessed. Application-specific online help can be defined to the CAS Help Manager using an application identifier of $MA and a function name of DEF*appl*, where *appl* is the name of the DEFLOGON for the application.

Broadcasts to MAI-FS Users

MAI-FS supports the use of broadcast facilities.

Broadcast messages are prepared for dispatch by entering /BCAST on any panel. The Broadcast Services : Primary Menu is presented, which displays a series of options that allow you to specify the broadcast text and select the category of user to receive the broadcast information.

If a broadcast is sent to one or more MAI users, MAI-FS forces a screen takeover (just as if the user had pressed a jump key). But instead of a jump taking place, an NCL procedure named $NMBRO is executed on behalf of the user. This NCL procedure can then display the relevant broadcast text on the user's terminal.

When $NMBRO processing completes, the MAI-FS session which was active when the screen takeover occurred is reinstated. A $NMBRO procedure is supplied that complements the Broadcast Services function. $NMBRO identifies the type of broadcast issued, determines whether the user will receive the broadcast information and, if so, presents the information on a full-screen panel. When the procedure terminates, normal MAI processing resumes.

Session Jumping Optimization

When a user jumps from one session to another, MAI-FS might have to perform a read buffer operation to retrieve the current screen contents from the terminal. As this operation can take a short time to complete for a remote terminal, it is helpful to minimize the operation as much as possible on this type of terminal. MAI-FS employs a number of techniques to do this operation.

Fast Jump Option

A fast jump option is available on the MAI Session details panel. If this option is specified, a jump away from the session results in a read buffer only if MAI-FS knows that the screen image has changed since the last read buffer was performed (that is, the last jump from the session), or since the last erase write data stream was saved. In other words, if MAI-FS has a current screen image and no input or output has occurred on the session, the read buffer need not be performed.

Note: If data is typed on to the screen for which MAI-FS believes it has the current image, and a jump is performed without the data being sent to the target application, MAI-FS proceeds to the next session, losing the data that was typed before the jump was requested.

Data Stream Save Option

MAI-FS can obtain the current screen image of a session in two ways. One, as described previously, is to perform a read buffer operation to the terminal. The other is for MAI-FS to remember an erase-write (EW) or erase-write-alternate (EWA) data stream received from the application. A data stream that begins with an EW or EWA command formats the whole screen, and therefore can be used to refresh the screen contents totally.

The Data Stream Save option controls the way in which MAI-FS remembers such data streams. If an application frequently uses EW or EWA data streams (for example, IMS or CICS), use of the appropriate option can improve jumping significantly. The option has the following possible values:

YES

Specifies that MAI remembers all EW or EWA data streams.

FASTJUMP

Specifies that EW or EWA data streams are only remembered for sessions which also specify the fast jump option.

NO

Specifies that EW or EWA data streams are never remembered.

For MAI-FS to remember the data streams, more storage must be used. The amounts involved depend upon the length of the data streams. The options provided allow you to minimize the storage used if necessary.

Note: Where supported, the storage resides above the 16-MB line.

This option is set using the MAIPARMS parameter group of the Customizer. The Data Stream Save option is on Page 3 of the parameter group.

Choice of Jump Methods

A number of methods are provided to enable you to perform a jump from one session to another:

- Press a nominated function (F) key.
- Press a nominated Program Attention (PA) key.
- Press the Attention (ATTN) key.
- Use a session skip command, which involves entering a command that begins with the session skip character (usually a period) and pressing the session skip key (usually F12 or F24). Session skip commands and keys are set using the MAIPARMS parameter group of the Customizer.

It is important to realize that the terminal hardware transmits no data when a PA key or the ATTN key is pressed, whereas when a function key is pressed any other data modified on the screen is sent along with the function key notification.

If the fast jump option is not in effect it is usually better to jump using function keys (either as a jump key or with a session skip command). MAI-FS could already have the current screen image and will be able to tell whether or not a read buffer is required by the presence or otherwise of modified data in the incoming data stream. If a PA key or the ATTN key is pressed and the fast jump option is not in effect, MAI-FS must always perform a read buffer.

Recommendations

The following recommendations apply if your system operates remote terminals and you want to minimize read buffer delays:

- Set the Data Stream Save Option to YES to ensure that MAI-FS remembers all EW or EWA data streams (as long as the storage overheads are acceptable). The exception is when only PA keys are used to jump and the fast jump option is not used. Under this circumstance a saved EW or EWA data stream is never used, as MAI-FS always performs a read buffer. Storage can be minimized if you set the Data Stream Save Option to FASTJUMP so that data streams are only saved for a session that also has the fast jump option.
- If the consideration for the fast jump option (see page 156) is acceptable, use the fast jump option on sessions operated from remote terminals.
- Have users use function key jumping or session skipping if the fast jump option is not used.

Chapter 13: Session Replay Facility

This section contains the following topics:

Overview

Help Desk personnel and systems programmers are often faced with the task of remotely diagnosing end-user problems. For instance, a user is having a particular problem with a CICS transaction, and must explain, usually over the telephone, what they are entering and the various responses being received from CICS. This remote diagnosis is prone to error and misinterpretation, as the person diagnosing the problem cannot see exactly what the end user is seeing. The problem may then have to be conveyed to the programming department, where once again it is open to misunderstandings.

The MAI Session Replay Facility (SRF) addresses these issues. SRF allows authorized personnel to capture to disk, complete session dialogs of any terminal connected to CA SOLVE:Access. SRF can trace terminals that are using any SOLVE functions, including MAI-FS.

In the example, if the user experiencing the problem with the CICS transaction was logged on to CICS through MAI-FS, Help Desk personnel can capture exactly what the user typed in, and the exact responses received from CICS. The capturing of data in this way can continue for as long as necessary, and the captured data can then be reviewed from any authorized terminal. This review process allows each screen interaction to be displayed exactly as the original user saw it. SRF also has an auto-replay function that automatically shows each screen input and output in sequence, with short delays between each.

SRF can also be used for training purposes, showing staff various application flows that have been captured previously without the need to use a live application, and also for regression testing of new versions of applications to verify that their processing is consistent with previous versions.

Database Maintenance

Over time, the SRF database will probably fill up. If this happens, a message is sent to all Monitor class OCS users and information about the error are sent to the activity log. SRF will then cease to log any further trace activity.

If the database is full, the following steps must be taken:

- From the SRF menus, delete all captured sessions that are no longer required. This operation is explained below.
- Close and unallocate the TRFILE database. To do this, update the TRFILE parameter group in Customizer to erase the data set name, and action the group.
- Copy the database by using the REPRO function of the IDCAMS utility.
- Delete and re-define the TRFILE database.
- Copy the database back into the newly-defined TRFILE by using REPRO.
- Reallocate and open the TRFILE database. To do this, update the TRFILE parameter group in Customizer to specify the data set name, and action the group.

Access SRF

To access the Session Replay Facility, enter /SNASRF at the command prompt.

The SRF Primary Menu appears.

The following options available on the menu.

- Start Session Capture
- Stop Session Capture
- List Captured Sessions
- List Session Capture Requests

Start Session Capture

Option 1 of the SRF Primary Menu is used to start a session capture for a given terminal.

To start a session capture for a terminal, specify the name of the terminal in the Resource Name field and select Option **1**.

The terminal may or may not be in session with CA SOLVE:Access at the time of the request. If it is not, the request waits until the terminal logs on. A message is written to Line 3 of the panel to indicate the success or failure of the start capture request.

Stop Session Capture

Option 2 of the SRF Primary Menu is used to stop a session capture for a given terminal.

To stop a session capture for a terminal, specify the name of the terminal in the Resource Name field and select Option **2**.

A message is written to Line 3 of the panel to indicate the success or otherwise of the request.

List Captured Sessions

Option 3 of the SRF Primary Menu is used to present a selection list of the captured sessions on the SRF database. A full or partial terminal name can be entered in the Resource Name field to restrict the list to those terminals whose names start with the supplied prefix. If no name is supplied, a list of all sessions displays.

To list captured sessions, complete the Resource Name field as required and select Option **3**.

The sessions on the Dialog Selection list are listed under the following headings:

From Date and Time

The date and time at which the capture was started.

To Date and Time

The date and time at which the capture was stopped.

To list the data streams captured for a session, enter **S** next to the session entry.

The Element List panel appears listing the captured data streams.

To delete a session, enter **D** next to the entry. Delete sessions that are no longer required so that space can be freed on the database.

Having selected a session, it is then possible to replay it either step-by-step or automatically so that each screen interaction displays one after the other.

List Session Capture Requests

Option 4 of the SRF Primary Menu is used to present a list of the terminals for which capture requests are current. A full or partial terminal name can be entered in the Resource Name field of the panel to restrict the list to those terminals whose names start with the supplied prefix. If no name is supplied a list of all current capture requests displays.

To list session capture requests, complete the Resource Name field and select Option **4**.

The requests on the Request List are listed under the following headings:

Nodename

Displays the name of the terminal for which a capture request is current.

Date and Time

Displays the date and time at which the capture request was entered.

Status

Displays the status of the capture request.

ACTIVE

Indicates that the terminal is currently connected to CA SOLVE:Access and data being captured.

PENDING

Indicates that the terminal is not currently connected to CA SOLVE:Access, but that as soon as it is, the session data will begin to be captured.

STOPPED

Indicates that the capture has been stopped by use of the P line command.

Message

Is an area for messages when entries are selected.

By placing a **P** next to any entry on the Request List, the session capture for that terminal can be stopped. By placing an **S** next to any entry, the session replay function for that terminal can be entered. The next section explains how this function is used.

Session Replay Using SRF Dialog Element List Panel

As described previously, Option 3 from the SRF Primary Menu provides a list of sessions that have been captured to the SRF database. When a session is selected, SRF presents a list of all the data streams captured for the session. This list can also be presented using selection of a session from Option 4. The Element List panel displays the following information:

Trace Date and Time

The date and time at which the data stream was captured.

Direction

Either Outbound, indicating the data stream was sent to the terminal, or Inbound, indicating the data stream was received from the terminal.

Description

The type of data stream sent to the terminal or application.

Note: For more information, press F1 (Help).

Replay in STEP Mode

A session replay can be started in what is termed STEP mode.

To replay a session in STEP mode

1. Enter **S** next to a data stream entry.

   The entry selected governs the point in the session at which the replay commences. That data stream is displayed on the screen as the original user saw it.

2. Press Enter.

   The next data stream in sequence is displayed.

3. Repeat Step 2 for as many data streams as are stored for this session that you want to review.

Replay in AUTO Mode

A session replay can be started in what is termed AUTO mode.

To replay a session in AUTO mode

1. Enter **A** next to a data stream.

   The entry selected governs the point in the session at which the replay commences. The session flow is then automatically presented from that point, with pauses occurring between each screen display.

2. Let the presentation continue until all captured data has been displayed, or terminate the presentation at any point by pressing the F3 function key.

Non-display Data

SRF preserves the integrity of non-display data. That is, if the user enters data into a non-display (zero intensity) field on a screen, that data will not be visible during session replay.

Scroll a Screen Image

The screen on which a session is being replayed can have different dimensions from the terminal whose session was captured. Under this circumstance, it may only be possible to display a part of the original screen image at any one time. However, SRF allows you to scroll around the screen image to move the window.

To scroll up the image, press F7.

To scroll down the image, press F8.

To scroll to the left of the image, press F10.

To scroll to the right of the image, press F11.

These same facilities can be used if the replay is performed while in split-screen mode, where the whole screen is not available for the display.

The scroll keys can be used when the terminal whose session was captured was wider than 80 columns. SRF only displays up to 80 columns of data, even if the reviewing terminal can display more. In this case, the left and right scroll keys enable you to see the whole image.

Display a Data Stream in Hexadecimal

Individual data streams can be displayed in hexadecimal.

To display a data stream in hexadecimal, enter **H** next to the required data stream entry.

A scrollable display of the data stream in hexadecimal and character appears.

Data Stream Analysis

After a data stream has been selected for display, SRF can analyze it and report on its content.

To display the analysis, press the F5 key while viewing the data stream. The following illustration shows an example of such an analysis. SRF breaks down the data stream into its constituent parts and explains the meanings of all control characters and field contents.

```
------------------------ SRF : 3270 Datastream Analysis -------------- 1 of 86
 Resource  : TERM01
 Time      : SAT 11-APR-2000 23.53.40
 Direction : OUTBND
 WRITE.
 SET BUFFER ADDRESS (000,000)
 START FIELD EXTENDED, PROTECTED, HIGH, SPD, COLOR=TURQUOIS
 SET BUFFER ADDRESS (000,021)
 START FIELD EXTENDED, PROTECTED, HIGH, SPD, COLOR=TURQUOISE
 SET BUFFER ADDRESS (000,060)
 START FIELD EXTENDED, PROTECTED, HIGH, SPD, COLOR=TURQUOISE
 SET BUFFER ADDRESS (001,000)
 START FIELD EXTENDED, PROTECTED, SKIP, DISPLAY, COLOR=GREEN
 SET BUFFER ADDRESS (001,014)
 START FIELD EXTENDED, PROTECTED, HIGH, SPD, COLOR=WHITE
 SET BUFFER ADDRESS (001,019)
 START FIELD EXTENDED, UNPROTECTED, HIGH, SPD
 INSERT CURSOR (001,020)
   DATA='1'
 REPEAT C' ' TO ADDRESS (001,032)
 START FIELD EXTENDED, PROTECTED, SKIP, DISPLAY, COLOR=GREEN
 SET BUFFER ADDRESS (002,000)
 START FIELD EXTENDED, PROTECTED, HIGH, SPD, COLOR=RED
   DATA='N11D08 TRACE FOR TERM01 STARTED, ACTIVATION PENDING.'
```

SRF Control Function Panel

Press the F6 key while reviewing a data stream to gain access to the SRF : Control Function panel which allows you to change various control options. The following fields are on this panel:

Partition ID

Specifies the number that identifies the terminal partition to which the session replay applies.

Some terminals have a facility known as hardware partitioning. This feature allows the application to divide up the terminal into a number of sections, or partitions. If SRF is used to capture data produced by such an application, the replay function displays, by default, the partition being addressed at the time.

Print Fields

Specifies whether you want to send a data stream analysis (like that produced by pressing the F5 key while viewing a data stream) to a VTAM printer. Enter YES in this field and the VTAM node name of the printer in the next field to send the analysis to the printer.

Note: Only LU Type 1 printers are supported for this function, and use of EASINET is required. Your EASINET NCL procedure must support LU Type 1 devices. The LU Type 1 code within the distributed EASINET NCL procedure $EASINET is adequate for support of SRF printing.

Printer Name

Specifies the name of the printer, as mentioned in the previous field description.

Playback Mode

Allows you to change the current playback mode. You can change between STEP and AUTO at any time.

Delay Inbnd

Specifies the delay in seconds that SRF is to impose after displaying an inbound data stream while replaying in AUTO mode.

Delay Outbnd

Specifies the delay in seconds that SRF is to impose after displaying an outbound data stream while replaying in AUTO mode.

Chapter 14: MAI Screen Image Services

This section contains the following topics:

Overview

MAI-SIS provide various facilities associated with an application screen image under MAI. These facilities allow an MAI user to store, redisplay, print, or transfer screen images or parts of screen images (using cut and paste facility) while in the MAI environment.

The use of MAI-SIS facilities is described in the *User Guide*.

MAI-SIS uses some of the SRF functions. For example, the redisplaying of stored screen images is under the control of SRF. MAI-SIS screen images are stored in the ACDB database. If this database is shared between multiple CA SOLVE:Access regions, the images can be stored on one region and viewed on another.

Note: For information about the ACDB database, see the *Installation Guide*.

MAI-SIS Facilities

All the facilities of MAI-SIS can either be selected from the MAI-SIS menu or executed, bypassing the menu, if the required option is known. For example, to store an image, select the Save Screen Into Image File option from the menu. Alternatively, by using the MAI-SIS session command prefix .P, and adding the required option, the menu can be bypassed (for example, .P1). The ability to bypass the MAI-SIS menu is very useful when there is a need for storing or printing more than one image in succession.

Note: You can also assign an MAI session jump key to jump to the MAI-SIS Menu.

Stored Screen Images

The currently displayed application screen image can be stored on the MAI-SIS database. The image is stored with information such as the storing user ID, terminal, current time, and description. All images are stored with the user ID as the record key and are only accessible by that user.

When a user stores the first screen image, a master record is created containing information about the stored image, and some physical attributes of the storing terminal (for example, screen dimensions). This information enables MAI-SIS to determine, when redisplaying an image, if the stored image fits into the display area of the viewing terminal.

For example, a user stores an image using a Model 4 type terminal which can have up to 43 lines in the display area. The user then decides to use a Model 2 type terminal which only supports up to 24 lines to redisplay the previously stored image. MAI-SIS caters for this situation and allows the user to scroll through the display. To support this facility, a master record is created per model type. Therefore, if a user has stored images using Model 2, 3, and 4 type terminals, three master records are created for that user.

A user can display previously stored screen images at any time. From the display list of stored images, a user can perform standard MAI-SIS functions such as printing or transfer of one or more images. If data stream analysis is required, screen images can also be displayed in their hexadecimal format.

Screen images can be deleted selectively or as a group (that is, by deleting the entire image queue).

Screen Image Printing

A user can print the current screen image or previously stored images. MAI-SIS supports screen image printing using the following:

- SOLVE Print Services Manager (PSM)
- VTAM Printer
- JES Printer
- JCL to JES Spool

When the print option is selected, the user is prompted for printer definition details such as the printer type and the printer name or destination. To enable the print function to be invoked while bypassing the MAI-SIS menu, the print details can be stored permanently using the F6 key from the menu panel.

Note: You can also assign an MAI session jump key to print the current screen.

If you use a JCL printer type, the printer name must be known to the Job Entry Subsystem (JES). Use the $SISJCL member, which provides a skeleton example for printing images using the JES subsystem, as a guide to implementing the JCL method of printing.

When sending screen images to the JES spool for printing, MAI-SIS submits an IEBGENER job to copy the image into the spool. To use this facility the supplied JCL for the job must be customized to suit installation standards. The JCL code is stored into variables in the MAI-SIS NCL procedure $SISJCL, which is distributed in the library ACTEXEC. This procedure must be modified before using this facility.

Important! Changing the command authority of the TERMINAL, OPNDST, CLSDST, ALLOC, UNALLOC, or UDBCTL commands can cause MAI-SIS to give unpredictable results. These commands are used in the $SISPRT NCL procedure. If users cannot have these commands executed on their behalf by MAI-SIS, MAI-SIS print functionality is inhibited.

Screen Image Transfer

Image transferal allows a user to send the current screen image or previously stored images to other MAI users. A warning panel can be sent ahead of the transferred images to advise the receiver as to who has sent the images and how many have been received. The receiving user has the option to view as many of the images as required and to terminate the viewing at any time.

To transfer screen images, users must have the appropriate UAMS authority level to issue an NSBRO command.

Screen Image Replacement Facility

The Screen Image Replacement Facility is a cut and paste facility. That is, a facility that allows a user to take some data from one application session panel and insert it into another panel.

You save the screen image that contains the data that you want to copy. The data is then stored in a string assignment queue. Each element of the queue is any part of a panel line or an arbitrarily defined window (in the panel whose data you want to copy from).

The elements of the queue are entered in an appropriate order into the fields of another panel.

In this way, data is quickly entered into the fields of a different panel and treated by the application as if the user had entered them.

Note: For a detailed description of how to use the Screen Image Replacement Facility, see the *User Guide*.

Screen Image Maintenance

The Administration menu contains a new option, S (Screen Image Services Maintenance), with a shortcut of /SISM. If you are authorized for MSDM, you can maintain the stored screen images for *all* users. For example, you can delete the images owned by a user who has left your organization.

Chapter 15: MAI-FS Operational Considerations

This section contains the following topics:

MAI-FS Presentation

MAI-FS is designed as a productivity tool for network users. While some users may require access to many of the CA SOLVE:Access Network Diagnosis, Administration and Definition options, as in the case of network operators and technical support staff, others may have no need for direct access to functions other than MAI-FS. It is important that the presentation and usability of MAI-FS be customized to provide as transparent an interface to these users as possible.

A number of techniques are available to simplify access to MAI-FS as discussed below.

Single Function Users

Network users requiring access to MAI-FS need SOLVE user IDs that authorize them for MAI privileges. If they have no requirement for access to other SOLVE facilities, they are termed single function users. When such a user logs on to CA SOLVE:Access, the system recognizes that the user ID has only one function and automatically places the user in the appropriate mode (in this case, MAI). When the user leaves that mode they are automatically logged off.

Definition of single function users is the simplest method of making MAI-FS facilities available to most network users. Used in conjunction with Stored Definitions and Automatic Session Establishment, single function user IDs can provide their owners with MAI-FS facilities without the user being aware of the other CA SOLVE:Access facilities at all.

Stored Definitions

All MAI-FS users can use stored definitions to provide a selection of potential sessions, whose attributes, jump keys and other parameters have been pre-defined and stored on the ACDB data set. These are listed on the MAI-FS Selection Menu and can be started, selected or terminated as required.

The MAI-FS Privilege Class associated with a user ID determines whether the user may start sessions other than those predefined for the user ID.

Automatic Session Establishment

Privilege class also determines the action that MAI will take on behalf of the user on entry to MAI mode. The following shows the session establishment action taken for each privilege class.

A

MAI-FS menu is presented. No sessions are opened. The user can open any Stored Definition session or start sessions not previously defined.

B

MAI-FS menu is presented. No sessions are opened. The user can open only those Stored Definition sessions listed on the MAI menu.

C

All Stored Definition sessions defined for the user are opened (if possible) and the screen is assigned to the first active session in the list. The MAI menu remains in the session circle.

D

All Stored Definition sessions defined for the user are opened (if possible) and the screen assigned to the first active session. The MAI menu is removed from the session circle.

The choice of Privilege Class therefore determines the manner in which MAI-FS is presented to the user, ranging from full function availability provided to Class A users, to complete control over session establishment for Class D users.

In the case of Class D users, consider the following, fairly typical, scenario:

- A (non-technical) network user requires access to a CICS enquiry function and an order-entry application. The usual procedure is to log on to CICS to perform enquiries, log off, then log on to the order-entry application. This process may take place frequently.
- By assigning the user a single function (MAI) user ID, with a privilege class D, and with two Stored Definition sessions defined, one for CICS and the other for the order-entry application this user can be provided with a fully customized MAI environment in which the two sessions are automatically created. The MAI menu is not displayed and the user can jump between the sessions with a single jump key. The user sees no other SOLVE facility and might in fact be unaware that SOLVE exists in the system at all.

Use of Variable Substitution

The logon request held in a stored logon definition can contain NCL system variables and global variables. These variables are substituted before any logon is attempted. The most useful of these variables in the case of MAI-FS are the &USERID and &USERPW system variables.

For example, a logon request in a stored definition could be:

```
TSO &USERID/&USERPW.
```

The user's user ID and password would then be substituted into the request, and as long as the user's TSO user ID and password are the same as their user ID and password the TSO session could complete without any further prompting from TSO.

MAI User Verification

The MAI Logon, Modify and Update functions allow the user to define a logon string to be passed to the application as initial logon data. The MAI User Verification function is invoked by these functions when the logon string contains the &USERPW system variable. A screen is presented containing a prompt for the user's password. Upon successful entry of the password the MAI session modify or update is completed. If the password is not correctly supplied within the retry limit (PWRETRY system parameter) or if the F3 or F4 keys are used to terminate the user verification then the associated MAI function is canceled (No updates to either the stored definition or the MAI session parameters are made.)

Notes for individual functions:

LOGON

If the logon string is modified and the string contains the &USERPW system variable then the MAI User Verification function will be invoked to validate the user.

UPDATE

If the logon string contains the &USERPW system variable then the MAI User Verification function will be invoked to validate the user.

MODIFY

If the logon string contains the &USERPW system variable then the MAI User Verification function will be invoked to validate the user.

Session End Panel

When an MAI-FS session ends, the user can be presented with a special panel. This panel, by default, contains information regarding the session that has ended, and instructions on how to jump away from the panel.

The Session End Panel (in the MAIPARMS parameter group in Customizer) governs whether a session-end panel is presented when a session ends, and if so, the panel format. Possible values are:

Full

Displays a panel containing information about the session that has ended, and instructions as to how to proceed to the next session. This is the default.

Short

Displays a panel that contains three asterisks (***) in the upper left corner.

None

No session end is presented. An automatic jump to the next session, or the MAI menu is performed.

Note: A full session end panel is always displayed if the session ends because of an error condition. This panel includes additional error messages as appropriate.

To specify this option

1. Enter **/PARMS** on any panel.
2. Enter **U** next to the $AC MAIPARMS parameter group in the Tuning category.
3. Use the F10 keys to go page 3.
4. Enter the required value in the Session End Panel field. Press F1 (Help) for help.
5. To action the parameter group, press F6 (Action).
6. To update the parameter group, press F3 (File).

Session Scripts

The use of session scripts (see page 121) can significantly improve the operation of MAI-FS sessions, especially when used to automate session start.

Session end scripts are always run in the BSYS environment. Therefore, the system variable &USERID contains the BSYS user ID (for example, NM22BSYS) and &USERPW contains blanks.

Session Protocols

An MAI-FS session functions as a true SNA LU-type adhering to the SNA protocols described in the IBM publication 3270 Component Description, or as an LU-Type 0, adhering to the protocols used by VTAM to support non-SNA 3270 devices.

SHOW MAI Command—Monitor MAI Sessions

The SHOW MAI command can be used to determine the session states of MAI sessions. The command displays the status of all MAI session currently held for either the entering user or all users.

Information given by the SHOW MAI LIST=SUMMARY includes:

- The session ID which identifies each session
- The status of each session
- The LU name in use for each session
- The region name for each session to show the relationship between physical LUs and MAI-FS sessions
- The average buffer storage for the listed sessions, displayed on a summary line

The additional information given by the SHOW MAI LIST=ALL is:

- The name of the DEFLOGON used
- The NCL process ID of a script, if active
- The amount of storage used for data stream and image buffers

When a session is not active, the status values displayed are:

ACBW

Waiting for ACB open.

BNDW

Waiting for BIND from an application.

UNBW

Waiting for UNBIND from an application.

TRMW

Waiting for ACB close (waiting for termination to complete).

Other status values are:

INB

In Bracket.

BETB

Between Brackets.

BBP

Begin Bracket Pending.

SEND

Send State (can send to application).

RCV

Receive State (cannot send).

DRWT

Waiting for a definite response (cannot send).

STBY

Standby State (can send) -Indeterminate state (state change in progress or not in session).

The CON field in the display can contain one of the following:

YES

Session established and available.

NO

Session not yet established.

LCK

Session established but session is locked because session state is not such that data can be sent (for example, MAI is awaiting a response from the application).

Activity Log Messages

Messages are written to the activity log each time an MAI-FS session starts or ends, indicating the user ID and terminal concerned, the name of the application with which the session is created and the VTAM ACB name used for the session.

Any errors that occur on a session are also logged.

Chapter 16: Setting Up the Initialization File

This section contains the following topics:

Generate an Initialization File

If you are deploying multiple regions, each region must be configured for its local environment. When you have configured your first region, you can build an initialization file from that region and then configure it for use with your other regions. This removes the need to customize each region with Customizer.

The tasks outlined below show how to configure a region from an initialization file. The initialization file is produced from a running region for your product.

To generate an initialization file

1. From the Primary Menu, enter **/CUSTOM**.

   The Customizer panel appears.

2. Select option G - Generate INI Procedure.

   The Customizer : Generate INI Procedure panel appears.

3. Enter the data set name and the member name of the file in the Generate INI File Details section.

   Note: The data set must be in the commands concatenation of the RUNSYSIN member for the region in which it is used.

4. Ensure that the member name and data set name are correct. Enter **YES** in the Replace Member? field if you are replacing an existing member.

5. Press F6 (Action).

   The initialization file is generated.

6. Make a note of the data set and member names and press F6 (Confirm).

   The details are saved.

Configure the Initialization File

The initialization file must be configured before it can be used on other systems. You can do this as follows:

- Configure an individual initialization file for each region.
- Configure a common initialization file for multiple regions.

You can use system variables and static system variables with both of these methods. The variables substitute for the initialization parameters in the INI file.

Configure a Common Initialization File

You can customize an initialization file using variables so that it can be used for multiple regions.

To configure a common initialization file

1. Create a data set that is available to every region to be initialized from the common initialization file, for example, PROD.INIFILES.
2. Add the newly created data set to the COMMANDS concatenation of the RUNSYSIN member to every region to be initialized from the common initialization file.

   Note: RUNSYSIN is located in TESTEXEC.
3. Copy the initialization file generated into the new INIFILES data set.
4. Use your TSO editing tool to open the initialization file in edit mode.

5. Replace the relevant generated variables in the initialization file with the following system variables:

   &ZDSNQLCL

   The local VSAM data set qualifier.

   &ZDSNQSHR

   The shared VSAM data set qualifier.

   &ZACBNAME

   The primary VTAM ACB name used by the region.

   &ZDSNQLNV

   The local non-VSAM data set qualifier.

   &ZDSNQSNV

   The shared non-VSAM data set qualifier.

   &ZNMDID

   The domain identifier.

   &ZNMSUP

   The system user prefix.

6. Replace the relevant generated variables in the initialization file with the z/OS static system symbols as follows:

   &SYSCLONE

   The short name for the system.

   &SYSNAME

   The name of the system.

   &SYSPLEX

   The name of the sysplex.

   &SYSR1

   The IPL VOLSER.

7. Save the changes to the initialization file.

Configure Individual Initialization Files

You can customize an initialization file generated from one region so that it can be used for another region.

To configure an individual initialization file for each region

1. Use your TSO editing tool to open the initialization file in edit mode.
2. Substitute the parameters in the initialization file with *one* of the following:
   - Hard-coded data set names for the region in which the file is used
   - System variables

   This enables the initialization file to work in regions with different data sets than the region in which it was generated.
3. Save the changes to the initialization file.
4. Copy the initialization file to the region's TESTEXEC or one of the other libraries in the COMMANDS concatenation.
5. Repeat steps 1 to 4 for each initialization file needed.

Note: The region from which the original initialization file was generated should have the same product sets as the destination regions that will use that initialization file.

Start Your Region from an Initialization File

The name of the initialization file must be specified by the INIFILE parameter in the RUNSYSIN member.

Updating your RUNSYSIN member causes your region to set its initialization parameters from the initialization file. All Customizer parameter settings are overwritten.

To update your RUNSYSIN member

1. Use a text editor to open your RUNSYSIN member.
2. Insert the line PPREF='INIFILE=*membername*' into your RUNSYSIN member.
3. Save the member.

Chapter 17: Customizing the MAI : Primary Menu

This section contains the following topics:

MAI Primary Menu

An NCL procedure, $MAIMENU, runs the Multiple Application Interface Primary Menu (MAI : Primary Menu). You can modify the menu to requirements of your site.

The MAI : Primary Menu as supplied consists of a title line, a command entry field, an error message line, a selection list of available session definitions and some general information. The use of the MAI : Primary Menu is described in the *User Guide*.

Some examples of things you might want your MAI : Primary Menu to do are:

- Display a menu screen layout specific to your organization.
- Display only those features to users who have authority to use them.
- Set up nested menu levels.
- Alter menu function key usage.

Use the $MAIMENU code as a model on which to base your own MAI : Primary Menu. Define the name of the procedure used to present the MAI : Primary Menu with the SYSPARMS MAIMENU=*procedure_name* command. The MAI : Primary Menu defined in this way is used systemwide.

Use the SHOW SYSPARMS = MAIMENU command to determine the name of the procedure currently driving the MAI : Primary Menu.

Primary Environment Modes

The MAI : Primary Menu NCL procedure is started either when the previous primary environment process terminates or when the user jumps from an active MAI-FS session to the menu.

CA SOLVE:Access has three primary environment modes:

- Base NCL mode. The primary menu NCL process controls processing. Line messages generated from NCL procedures are queued for display once the primary menu NCL procedure terminates.
- OCS mode. Line messages are displayed under the control of OCS. The MSGPROC facility can be used to process message flows.
- MAI Full Screen mode. The MAI menu procedure controls the definition and selection of MAI-FS sessions. Line messages are queued until the procedure terminates. MAI-FS session commands are executed when the procedure terminates.

The SETMODE command controls how these modes are invoked. Mode changes are effected upon termination of the primary environment manager. Whenever OCS or MAI-FS modes terminate, the primary menu NCL procedure is re-invoked in the base NCL mode.

Commands, Verbs and Variables Used to Customize $MAIMENU

This section provides information about all MAI commands, verbs and other objects used to customize the MAI : Primary Menu.

SETMODE Command—Set Primary Environment Mode

The SETMODE command changes the primary environment processor.

This command has the following format:

```
SETMODE MAI or SETMODE OCS
```

Use this command to set the primary environment process to be one of MAI-FS mode, OCS mode or base NCL mode. Upon termination of the current primary environment, the requested primary environment processor is invoked. When OCS mode is specified then the Operator Console Services function is initiated. When MAI-FS mode is specified then the MAI : Primary Menu processor is invoked.

The primary menu procedure uses this command to select a different primary environment processor. When MAI-FS mode is active, the NCL procedure specified by the MAIMENU system parameter is executed. This procedure can then use the &MAICMD and &MAISCMD verbs to initiate the various MAI-specific functions.

NCL Verbs

This section lists and briefly describes the NCL verbs used to direct the MAI : Primary Menu's processing.

Note: For a complete description of these verbs, see the *Network Control Language Reference Guide*.

&MAISCMD Verb—Issue Session Command

This verb specifies an MAI session command against the current session, set by the &MAISGET verb. The values are processed in the order of the in-storage MAI selection list when the MAI procedure terminates.

This verb has the following format:

```
&MAISCMD {session-command}
```

Valid session commands values are:

S

Starts the session (if not already started) and selects it for display if no other has been previously selected.

A

Activates the session if not already active.

Note: The session is not selected for display.)

B

Positions the session as the last in the selection list of sessions.

T

Positions the session as the first in the selection list of sessions.

C, CU, CC, CF

Cancels the active session. The session failure can be reported to the associated application as either an unconditional, conditional, or forced disconnection.

H

Marks the session (if active) as hidden. The session is not selected using nonspecific session jumping.

SL

Marks the session (if active) as hidden until output arrives. Upon arrival of any data stream from the associated application, the hidden status is reset and the session is eligible for selection by nonspecific session jumping.

M

Modifies the session definition. If the session is active, certain attributes cannot be changed (for example, node name or script procedure).

U

Updates the session definition on external storage. A full-screen function is invoked to display the current stored definition. Changes made are not reflected in the current in-storage definition. This function can delete the session definition.

P

Purges the definition from the in-storage session definition list. The session is no longer displayed by the MAI menu. (However, unless it is also deleted using a stored definition update, it reappears when MAI is reinitialized.)

L

Logs on to an inactive session displaying the session characteristics (for modification or confirmation) before starting the session.

R

Repeats a session definition. The current session definition is used as a model to build a new session definition that is placed in the session list immediately after the one selected. The session ID of the new session definition is set to blanks

&MAICMD Verb—Issue Primary Menu Command

This verb specifies an MAI : Primary Menu command.

This verb has the following format:

```
&MAICMD {END|JF|JR|J}
```

The following values can be specified for the command string:

END

Requests termination of MAI as the primary environment processor. When the MAI menu process terminates, then control is passed to a new primary environment - either the primary menu or OCS.

JF, JR, J

Specifies to perform a jump when the MAI : Primary menu terminates.

&MAISGET Verb—Retrieve Session Information

This verb retrieves information about the specified session into user variables. This session is marked as the current session.

This verb has the following format:

```
&MAISGET { ID=sessid | RELNUM=nn | DEFAULTS }
         [ PREFIX={ MAI | prefix }]
         [ ATTR={ ANY | LIST }]
         [ FIELDS={ name |( name1,name2,...,namen )}]
         [ PAD ]
```

sessid is the one- to eight-character session identifier that uniquely identifies a defined MAI-FS session. A relative session within the defined list can be selected by supplying its numerical position with the RELNUM operand.

The DEFAULTS operand can be used to retrieve the default MAI-FS session characteristics. These values are set using the MAI Session Defaults panel (/ACADMIN.D) and are stored on the ACDB file.

&MAISADD Verb—Add Session

This verb adds a session definition based on user variables. The variables are the same as those set by &MAISGET.

This verb has the following format:

```
&MAISADD [ PREFIX={ MAI | prefix }]
         [ FIELDS={ name |( name1,name2,...,namen )}]
```

&MAISPUT Verb—Update Session List

This verb updates session list entries.

This verb has the following format:

```
&MAISPUT { ID=sessid | RELNUM=nn | DEFAULTS }
         [ PREFIX={ MAI | prefix }]
         [ FIELDS={ name |( name1,name2,...,namen )}]
```

System Variables

&MAIMNFMT

Returns the current menu format as long or short.

&MAI#SESS or &MAINSESS

Returns the number of currently defined sessions.

&ZMAIACT# or &ZMAIACTN

The number of active sessions associated with the current window.

&MAISKIPP

Set to the system-wide value for the session command prefix character.

&MAISKPK1

Set to the session command function key 1 (for example, F12).

&MAISKPK2

Set to the session command function key 2 (for example, F24).

&MAISCANL

Set to the scan limit for session commands.

&MAITITLE

The title that displays at the top of the MAI-FS main menu.

&MAISID

The ID of the current MAI session.

&MAIAE

Set to YES or NO to indicate the availability of the A and E primary commands.

Chapter 18: NSI Support

This section contains the following topics:

Access NSI Support

NSI support allows you to monitor sessions that are connected to MAI, using IBM Tivoli NetView® Performance Monitor (NPM).

Note: For information about NPM functions, see IBM's *NPM Installation and Customization Guide*.

To access NSI support, enter the **/ACADMIN.I** panel path on any panel.

The NetView Synergy Interface Support Menu displays. The following functions are available from this menu:

D

Display or Update the Status of MAI:NSI Support

S

Browse MAI:NSI Event Statistics

Update NSI Support Status

The NSI Support Status and Defaults panel displays the status of NSI support, and from this panel you can dynamically start and stop the NSI support function.

To update the configuration and status of NSI support, select Option **D** from the NetView Synergy Interface Support Menu.

The NSI Support Status and Defaults panel appears.

Display Event Statistics

The NSI Event Statistics panel displays the status of NSI support, the time started, and statistics of the events processed.

To display event statistics, select Option **S** from the NetView Synergy Interface Support Menu.

The NSI Event Statistics panel appears.

How NSI Support Operates

NPM correlates data received from session managers. Session managers pass event data to NPM using NSI. NSI is interfaced by a calling module called FNMNSI which is supplied with NPM.

When a terminal logs on to an application using the session manager, two physical sessions exist-the session between the terminal and the session manager, and the session between the session manager and the application. NPM uses the information passed from the session manager using the FNMNSI module, to correlate these two physical sessions. It displays and reports on them as one relay session.

FNMNSI is called when session manager events occur, for example, a session start, stop, or skip, or a logon or logoff. Information about the user, session manager, and sessions, as well as LU names, and session CIDs, is passed to FNMNSI.

The implementation of the MAI interface to NSI has three steps:

- MAI issues system events using Event Distribution Services (EDS).
- A long running NCL procedure is profiled for the MAI system events. It reads these events, and converts them to a format which is understood by the NSI. It then passes them to a SOLVE subsystem.
- The SOLVE subsystem builds a request vector and calls the FNMNSI module, passing the request vector.

NSI Events

The NSI events map well into NetView Access behavior, although all are not applicable to MAI. Therefore, only a subset of the NSI events has been supported in the SOLVE implementation. The following list describes all NSI events:

X'01'

Logon to Session Manager

X'02'

Reconnect to Session Manager

X'03'

Change selected Application Session

X'04'

Return to Session Manager from Application

X'05'

Logoff from Session Manager

X'06'

Disconnect from Session Manager

X'07'

Logoff from Session Manager (with pass)

X'08'

Disconnect from Session Manager (with pass)

X'09'

Application Session establishment

X'0A'

Application Session termination

X'0B'

Application Session establishment (with pass)

X'0C'

Application Session termination (with pass)

X'0D'

Logon or reconnect to Application in direct mode

X'0E'

Logoff or disconnect from Application in direct mode

The following events are applicable to the SOLVE implementation:

X'01'

Enter MAI for the first time

X'02'

Reconnect to a SOLVE session for a user who has already entered the MAI Primary Menu before disconnecting

X'03'

MAI session skip-from one application session to another

X'04'

Skip to MAI Primary Menu from an active application session

X'05'

Log off from CA SOLVE:Access for a user who used MAI while logged on

X'06'

Disconnect from CA SOLVE:Access for a user who used MAI while logged on

X'07'

Not applicable to MAI

X'08'

Not applicable to MAI

X'09'

Connection to Application from MAI Primary Menu

X'0A'

Terminate application session that was started from the MAI

Primary Menu

X'0B'

Not applicable to MAI

X'0C'

Not applicable to MAI

X'0D'

Not supported in MAI

X'0E'

Not supported in MAI

Implementation Overview

The following diagram illustrates how to implement the interface between CA SOLVE:Access and NSI.

- MAI issues system events using EDS when any of the applicable NPM session manager events occur.
- A long running NCL procedure receives and process these events as they arrive.
- The procedure formats the information into the appropriate structure for the call to the FNMNSI module and passes a session event vector to an assembler subtask.
- The assembler subtask builds a request vector that contains the embedded session event vector and calls the FNMNSI module, passing the event vector.

MAI System Events

A set of system events is generated by MAI. These events are distributed using EDS. An event is issued when any of the supported NSI events happens within a SOLVE region where MAI is running.

The following shows the EDS system events that are generated by MAI when NSI support is active:

$$SYS.MAI.LOGON

Logon to MAI.

$$SYS.MAI.RECONNECT

Reconnect to MAI.

$$SYS.MAI.SKIP.TO.SESS

Skip to a session in MAI.

$$SYS.MAI.SKIP.TO.MENU

Skip to the MAI Primary Menu.

$$SYS.MAI.LOGOFF

Logoff from MAI.

$$SYS.MAI.DISCONNECT

Disconnect from MAI.

$$SYS.MAI.SESSION.START

Start application session through MAI.

$$SYS.MAI.SESSION.STOP

Stop application session through MAI.

How MAI System Events Are Started or Stopped

The SYSPARMS MAIEVENT command setting controls whether MAI system events are issued. When the MAI:NSI interface is started, it automatically issues a SYSPARMS MAIEVENT=YES command to start these events.

If you want to turn off MAI events to stop the flow of information across the MAI:NSI interface to NPM, issue a SYSPARMS MAIEVENT=NO command. Stopping these MAI events while the MAI:NSI interface is active has the same effect as stopping the interface altogether, as the subsystem receives no information to send to NPM. If the interface is kept active, the events can be turned on to start sending information to NPM again.

Default Startup Options

You can set the default initial state of the MAI:NSI interface when CA SOLVE:Access initializes by changing an option on the MAI:NSI Support Status and Defaults panel.

Enter **YES** in the MAI:NSI Support Active? field to cause the MAI:NSI component to start automatically when CA SOLVE:Access initializes. After the MAI:NSI interface is active, you can use the SYSPARMS MAIEVENT command to control the flow of information through the interface.

Chapter 19: Implementing Activity Logs

This section contains the following topics:

Activity Logs

The activity logging facility records all the activity in your region. You can use the activity logs to help determine the cause of problems.

Two separate activity log formats exist:

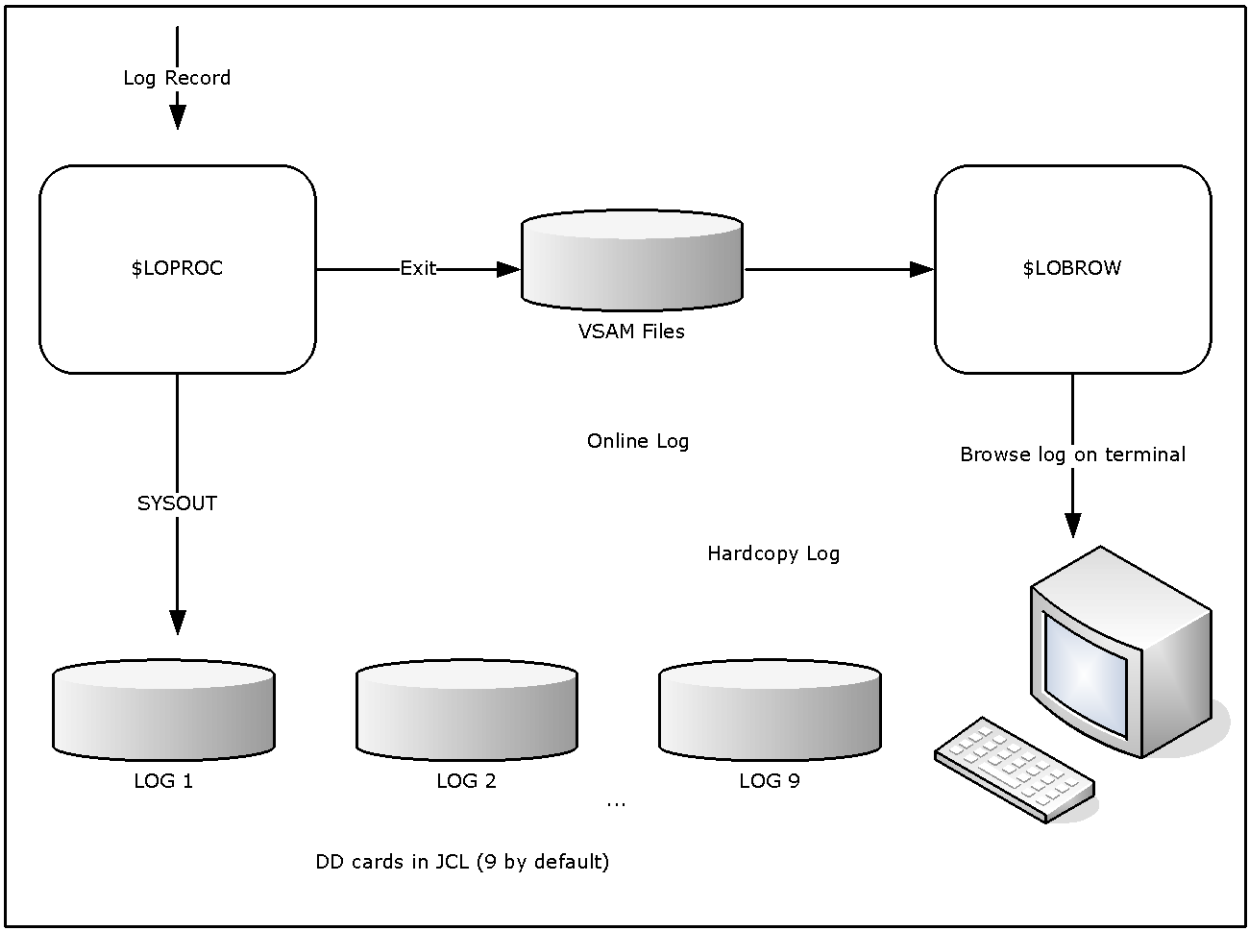
- Online
- Hardcopy

Log records are written to both formats.

By default, activity logs contain the following information:

- All commands entered
- All responses to commands entered
- Any unsolicited messages received from VTAM or the operating system, provided the related interfaces are available
- All messages explicitly written to the log by NCL procedures

The following illustration shows the path that the log record takes in the system.



The online activity log is supplied by the distributed procedure $LOPROC. The $LOPROC procedure writes log data to VSAM files (three by default). The VSAM files are accessed by a second procedure, $LOBROW, which allows online browsing of the log.

Note: $LOPROC and $LOBROW are the default procedure names. You can change these names by using the LOGFILES parameter group in Customizer (**/PARMS**).

Implement Online Activity Logging

During initialization, the region allocates, by default, three VSAM log files for online logging. However, you can allocate up to seven files.

Note: The log file IDs are of the form NMLOG*nn* and the data set names are of the form *dsnpref.rname*.NMLOG*nn*. (*dsnpref* is the data set prefix used during product installation and *rname* is the name of the region.)

Use Additional Log Files

If you want to make more than three files available to the region, define the new VSAM files and then customize the LOGFILES parameter group by defining additional logging data sets.

To customize the LOGFILES parameter group

1. Enter **/PARMS** at the prompt.

   The Customizer : Parameter Groups list appears.

2. Enter **U** beside the LOGFILES parameter group.

   The Customizer : Parameter Group panel for the LOGFILES parameter group appears.

3. Press F8 (Forward) to display the next page.

4. Complete the fields for each file you want to make available. To allocate more files, press F8 (Forward) again.

5. When you have specified the required number of log files, press F6 (Action) to allocate and open the files.

6. Press F6 (Action).

   The changes are applied.

7. Press F3 (File).

   The changes are saved.

Note: For more information about using this panel, press F1 (Help).

Administer Online Activity Log Files

From the Activity Log : Administration menu, you can do the following:

- Swap active activity logs
- List all days contained in log files and browse logs for a particular date
- List all log files and browse a particular file

To administer online activity log files

1. Enter **/LOADMIN** at the prompt.

   The Activity Log : Administration menu appears.

Note: For information about the options available on this menu, press F1 (Help).

Swap the Online Log

The online activity log automatically swaps to a fresh VSAM file when each file fills up.

You can manually swap your currently active VSAM file if you want to free a particular log file (for example, for backups).

Important! Swapping the current VSAM log causes the $LOPROC procedure to write all subsequent activity log records to the next VSAM log. If this log was previously used, it is reset. Therefore, you can no longer browse the old records that it contained.

To swap the online activity log

1. Enter **/LOGSWAP** at the prompt.

   The Activity Log Services : Confirm Swap Log panel appears.

2. Press F6 to request the log swap, or F12 to cancel your request.

   Note: If the $LOPROC procedure encounters a VSAM error when it is logging activity to an online log file, it automatically swaps to the next log file.

Use a Log Exit for the Online Log

You can create an NCL procedure to intercept, analyze, and react to the messages that are sent to the activity log.

Use the LOGFILES parameter group in Customizer to specify the name of your exit.

The exit is executed every time a message is sent to the log. Using the exit to perform complex functions can degrade the performance of the region.

Note: Ensure that your log exit procedure is well-tested before you put it into production.

Variables Available to the Activity Log Exit

The following variables are available to the activity log exit:

&#LO$RECORD

Contains records of the following formats:

time_generated user_id terminal_id message_text

The text of the message starts at the fourth word of the record.

arrival_time origin region* $$AOMTIME$$ *aom_time message_text

The text of the message starts at the sixth word of the record. This format lets you identify AOM-sourced messages.

You can change the contents of this variable. To suppress the message from the log, set the variable to NOLOG.

Note: For more information, see the &LOGREAD verb in the *Network Control Language Reference Guide*.

$LOG

Specifies a Mapped Data Object (MDO) that contains the message attributes. The MDO is mapped by the $MSG map.

You can use the &ASSIGN verb to query the MDO.

Note: For information about querying MDO components and additional variables, see the *Network Control Language Programming Guide*.

Example: Remove Messages from the NCL Log

The following shows an example procedure:

```
&CONTROL
-*-----------------------------------------------*
-* TO REMOVE IKJ56247I MESSAGES FROM THE NCL LOG. *
-*-----------------------------------------------*
&PARSE DELIM=' ' VARS=#LO$WORD* DATA=&#LO$RECORD
&IF .&#LO$WORD4 EQ .IKJ56247I &THEN +
        &#LO$RECORD = NOLOG
```

Enable the Log Exit

To enable the log exit

1. Enter **/PARMS** at the prompt.

   The Customizer : Parameter Groups list appears.

2. Enter **U** beside the LOGFILES parameter group.

   The Customizer : Parameter Group panel for the LOGFILES parameter group appears.

3. Enter the name of your activity log exit in the Log Exit Name field.

4. Press F6 (Action).

   The changes are applied.

5. Press F3 (File).

   The changes are saved.

Replace Your Online Logging Procedure

The default online logging procedure is $LOPROC. This procedure is designed to work in conjunction with the online browsing procedure $LOBROW.

You can replace the $LOPROC and $LOBROW procedures with your own customized NCL procedures. Alternatively, you can write a customized log browsing procedure to present the supplied data files (from $LOPROC) in your own format.

Write a Log Browsing Procedure

To write your own customized NCL procedure to replace $LOBROW, use the &FILE OPEN statement with FORMAT=DELIMITED.

The physical file structure of the supplied log files (NMLOG01, NMLOG02, and NMLOG03) is as follows:

Key Format

YYYYMMDDHHMMSSHS*nnnn*

nnnn=1000 + (reset every 100th of a second) and key length=20 bytes

Record Contents

ORIGIN

Terminal name

REGION

User ID

TEXT

Message text to display in the activity log

MSGATTR

2-byte color/highlight indicator. Colors are R=red, Y=yellow, W=white, B=blue, G=green, T=turquoise, or P=pink. Highlight values are R=reverse, B=blink, U=underscore, or N=none.

ORIGTIM

Remote domain time

ORIGDMN

Originating domain name

ORIGSRC

Remote terminal ID

Note: For more information, see the following references:

- The description of the &FILE OPEN verb in the *Network Control Language Reference Guide*.
- The *Network Control Language Programming Guide*.

Write Logging and Browsing Procedures

You can store your log records in whatever file format you want. If you do this, your log browsing procedure must match this file format.

Note: For more information, see the descriptions of the following verbs in the *Network Control Language Reference Guide*:

- &LOGREAD
- &LOGCONT
- &LOGDEL

Implement Logging and Browsing Procedures

If you write your own browsing procedure or your own logging and browsing procedures, you need to implement them.

To implement your procedures, update the LOGFILES parameter group in Customizer with your parameter names and then action the group.

Hardcopy Activity Log

A region can have more than one hardcopy activity log, of which only one is open for logging.

Your region can be configured to perform logging to disk, tape, or hard copy. From one to nine logs can be specified by including the required number of DD statements in the execution JCL. Logging can be specified to wrap when the last log is full or is swapped.

To obtain the status of these logs, use the SHOW LOGS command.

Note: When logging to disk the following DCB attributes should be used:

```
DSORG=PS,RECFM=VBA,LRECL=137,BLKSIZE=15476
```

Format of Logged Information

Each entry recorded on the log is in the following format:

```
12.04.23.12   SMITH     TERM54    +V NET,ACT,ID=NCP001
```

This entry consists of the following information:

- A time stamp in the format *hh*.*mm*.*ss*.*hs* (where *hh* is the hour, *mm* is the minute, *ss* is the second, and *hs* is the hundredth of a second)
- The user ID that entered the command or logged the message
- The terminal from which the command was entered or to which a message is sent
- The text of the message or command

Commands are highlighted with a plus sign (+) prefixed to the text to make it easier to distinguish commands from messages when browsing the log. If the command entered is an unsolicited VTAM command, it is highlighted and prefixed with an equals sign (=).

Format of Logged Timer-initiated Commands

Commands that are executed as the result of a timer-initiated command are prefixed by a plus sign, followed by the identity number of the timer command responsible. This is in the format *#nnnn*.

Example: Logged Timer-initiated Command

```
15.00.00.01   NETOPER   CNTL01    +#0005 D BFRUSE
```

Format of Logged Commands Executed in Background Environments

Commands executed under the control of background environments are identified by the following keywords in the user ID field for the command text and any resulting messages:

BG-SYS

Background System Processor

BG-MON

Background Monitor

BG-LOG

Background Logger

Format of Logged Commands from NCL Procedure-dependent Environment

If a command is executed from an NCL procedure-dependent environment (&INTCMD), the node field on the log contains the NCL ID of the process issuing the command.

Format of the Hardcopy Log

The hardcopy log data set has the following format:

- A heading on each page—contains the day and date on which the log was created and the system identifier (NMID) of the originating region.
- A log identifier on the right side of the page. The log identifier is the ddname under which the log was created. This log identifier assists log collation after printing.
- 60 lines on each page—this format can be altered to suit your requirements using the SYSPARMS LOGPAGE operand.

  Note: For information about LOGPAGE, see the *Reference Guide*.

Swap the Hardcopy Log

Swapping the current log frees the log for immediate printing. To swap the log, use the LOGSWAP command. Swapping the log is possible only when another unused log remains to which logging can continue. You can specify up to nine logs. Logs do not need to be consecutive.

When a log is swapped, the log status, the requesting user ID, and the reason for the swap are recorded. You can display these details with the SHOW LOGS command.

Each of the logs is identified in the JCL by the LOG DD name followed by a single digit in the range one to nine.

Example: Log Name

```
//LOG4     DD    SYSOUT=A,FREE=CLOSE
```

Mixing of device types is also valid. Inclusion of FREE=CLOSE prints the log when it is released by the LOGSWAP command.

Wrap the Hardcopy Log Data Sets

Wrapping lets you reuse a LOG data set when all of the available LOG data sets have been used.

The LOGWRAP SYSPARM determines whether log data set wrapping is allowed. You set the value of this SYSPARM in the Are Activity Logs to Wrap? field when you customize the LOGFILES parameter group in Customizer (**/PARMS**).

If you specify NO (the default) in the Are Activity Logs to Wrap? field, then wrapping is not permitted. When all the LOG data sets have been used due to successive LOGSWAP commands, the previous LOG data sets cannot be reused. After the last LOG data set is used, any further LOGSWAP commands are rejected.

If you specify YES in the Are Activity Logs to Wrap? field, log wrapping is allowed according to the following rules:

- If you are directing your LOG data sets to SYSOUT, then, as each LOG*n* DD card is used, the data set is dynamically unallocated as a result of the FREE=CLOSE option. In this case, you can reissue an ALLOC command to reallocate another SYSOUT file under the same DD name. For example:

  ```
  ALLOC DD=LOG3 SYSOUT=A FREE=CLOSE
  ```

  This DD name is now available for use as another LOG data set. Subsequent LOGSWAP operations can now reuse this LOG data set rather than rejecting the command when the last LOG data set is used.
- If YES is specified but the LOG DD cards point to sequential data sets, log wrapping overwrites the earlier LOG data held in these data sets. You should take precautions to archive the existing data before allowing the wrap to occur.

Cross-Reference Hardcopy Logs

To make it easier for operations staff to piece the full log together, certain information is recorded on the last and first lines of LOG data sets that have been swapped.

The first line of a new log that is used in place of a swapped log contains the reason for the swap, or the initiating user ID.

The last message printed on a swapped log is the DD name of the new log. Also printed at the start of the new log is the DD name or logical ID for the previous log.

I/O Errors on the Hardcopy Log

If an I/O error occurs on a log, the log is closed and the next available log is automatically swapped to, if one is available, and logging continues. This also applies to data set full conditions when logging to disk.

If the I/O error occurs on the last available log, a warning message is sent to all monitor terminals informing them that logging has ceased. The STATUS command also includes a warning message if logging is stopped. All log messages are passed to LOGPROC for analysis even if no log output is possible.

Write to the System Log

The SYSPARMS SYSLOG operand can be used to direct your region to write all logged output to the system log and to its own log, or to write all VTAM PPO messages received to the system log.

Note: For information about the SYSPARMS SYSLOG operand, see the *Reference Guide*.

Chapter 20: Implementing Print Services

This section contains the following topics:

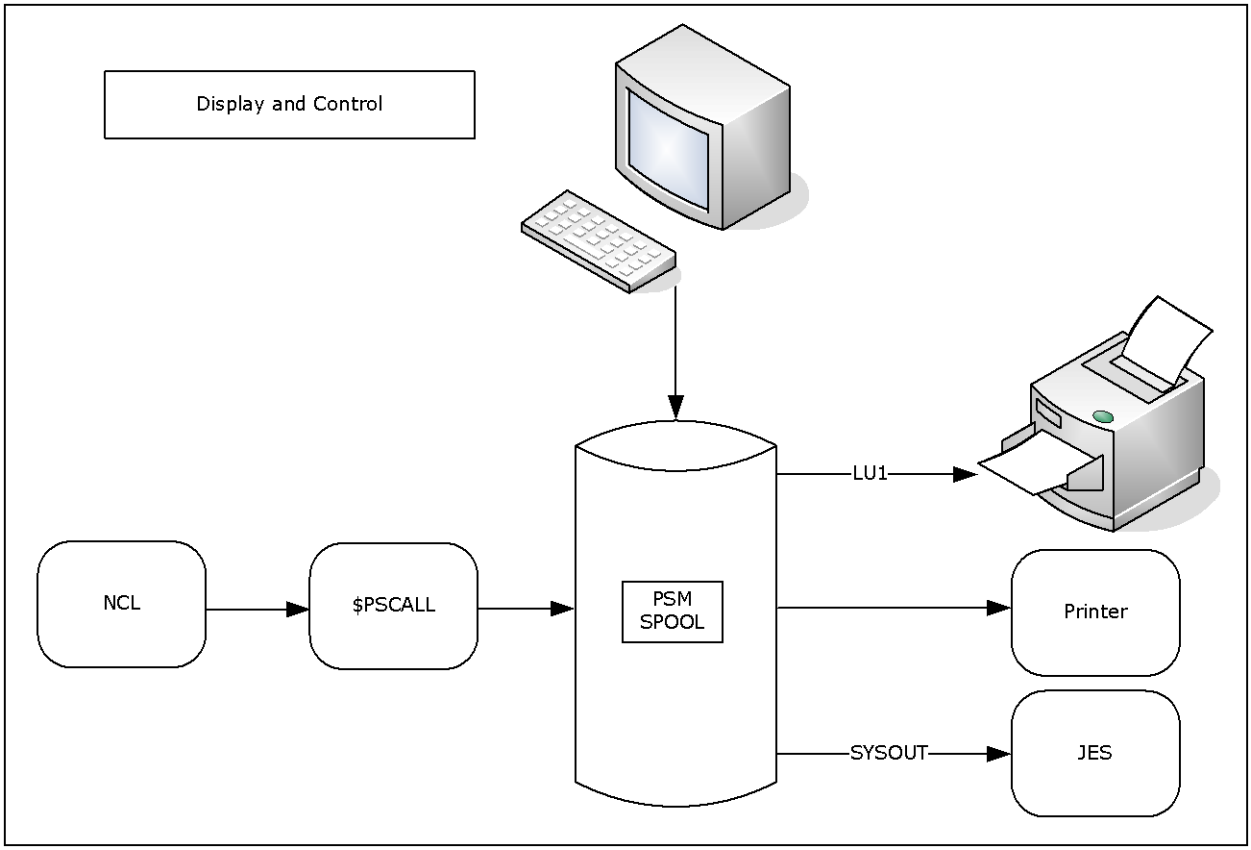
Print Services Manager

Print Services Manager (PSM) allows you to specify the format of a print request and on which printer it is printed. Print requests can be viewed online before or after printing and can be redirected to files rather than printers.

PSM provides the following features, which can be customized to suit your requirements:

- Printer definition facilities
- Form definition maintenance
- Setup definition maintenance
- Default printer assignment maintenance
- Alias printer name definitions
- Banner page customization on output
- Spooled print request browsing, retention, and redirection to a different printer
- Integration with NCL-based components

The following illustration shows the different ways that PSM can be used to control printing requirements.



Access PSM

The customizable functions of PSM are accessed from the PSM : Primary Menu.

To access PSM, enter **/PSM** at the prompt.

Note: You can also access PSM directly by invoking the $PSCALL NCL procedure from OCS or an installation written NCL procedure. The PSM NCL interface is described in the *Network Control Language Reference Guide*.

Add a Printer Definition

A printer definition defines where, how, and on what paper output is printed. A printer definition is required for each printer at which output is printed.

To add a printer definition

1. Enter **/PSMPRTR** at the prompt.

   The PSM : Printer Definition List appears.

2. Press F4 (Add).

   The PSM : Printer Definition panel appears.

3. Complete the fields, as required.

   Note: For information about the fields, press F1 (Help).

4. Press F3 (File).

   The definition is saved.

List Printer Definitions

You can display a list of all the printer definitions defined for your region. This lets you browse and perform maintenance on the listed definitions.

To list all printer definitions, enter **/PSMPRTR** at the prompt.

Add a Form Definition

A form definition is required for each type of paper on which output is printed. The Form Definition Menu is used to set up and administer these form definitions.

To add a form definition

1. Enter **/PSMFORM** at the prompt.

   The PSM : Form Definition List appears.

2. Press F4 (Add).

   The PSM : Form Definition panel appears.

3. Complete the fields and press F3 (File).

   The form definition is saved.

   Note: For information about the fields, press F1 (Help).

List Form Definitions

You can list all of the form definitions defined for your region and then browse and perform maintenance on them.

To list all form definitions, enter **/PSMFORM** at the prompt.

Add Control Characters

Control characters are sent to a printer before or after (or both) the output is printed. They are defined in setup definitions.

To add control characters

1. Enter **/PSMSET** at the prompt.

   The PSM : Setup Definition List appears.

2. Press F4 (Add).

   The PSM : Setup Definition panel appears. To access the second panel of the setup definition, press F8 (Forward).

   Complete the fields, as required.

   Note: For information about the fields, press F1 (Help).

3. Press F3 (File).

   The setup definition is saved.

List Control Characters

You can display a list of all the setup definitions defined for your region. This list lets you browse and perform maintenance on the listed definitions.

To list control characters, enter **/PSMSET** at the prompt.

Add a Default Printer for a User ID

Each user ID in your region can be assigned a default printer. Default printer assignments let you define the printer to which output is sent whenever a user ID does not specify a printer.

To add a default printer for a user ID

1. Enter **/PSMDFTP** at the prompt.

   The PSM : Default Printer Assignment List appears.

2. Press F4 (Add).

   The PSM : Default Printer Assignment panel appears.

3. Complete the following fields:

   User ID

   Specifies the User ID of the user to whom the printer is assigned a default.

   Printer Name

   Specifies the name of the printer to which this user's printing is sent.

   Press F3 (File).

   The default printer assignment is saved.

List Default Printers

You can display a list of all the default printer assignments defined for each user ID. This list lets you browse and perform maintenance on the listed definitions.

To list default printers, enter **/PSMDFTP** at the prompt.

Clear the Printer Spool

Print requests are retained on the print spool if an error occurs during printing or if HELD is specified on the PSM : Print Request panel. The PSM clear spool panel is used to clear print requests from the print queue.

Note: This function is available to authorized users only.

To clear the print spool

1. Enter **/PSMADMN** at the prompt.

   The PSM : Administration Menu appears.

2. Enter **CS** at the prompt.

   The PSM : Clear Spool panel appears.

3. Complete the following field:

   Date

   Specifies that all print requests added to the spool before or on this date are deleted.

   Press F6 (Action).

   The print requests are deleted.

Send Print Requests to a Data Set

Two printer exit procedures are distributed with your product. Each writes the output for a print request to a data set. The procedure $PSDS81X can be customized to specific site requirements. The procedure $PSDS81Z offers the same functionality with improved performance, but cannot be customized. The target data sets for both procedures can be sequential or partitioned.

Parameters that control the operation of the exit are defined in the Exit Data portion of the printer definition. Procedures that pass data to PSM for printing have the ability to override the exit data specified in the PSM printer definition.

The procedures use the parameters contained in the exit data to do the following:

- Determine the target data set
- Determine how to process a data line with a skip amount of zero
- Set the length of the lines print

How the Procedures Process a Print Request

The procedures read each line of print data and write it directly to the nominated data set. Each print line is analyzed according to skip control before processing. This continues until all lines of data for the print request have been received from PSM and written to the nominated data set.

$PSDS81X and $PSDS81Z Parameters

The $PSDS81X and $PSDS81Z parameters, which are coded as keyword parameters, are as follows:

```
  DSN=datasetname
[ DISP={ SHR | OLD | NEW | MOD } ]
[ LRECL={ n | 80 } ]
[ SKIP0={ NEWLINE | DISCARD | DESTRUCTIVE |
          NONDESTRUCTIVE } ]
[ CYL= pri [,sec] [,dir] ]
[ TRK= { pri [,sec ] [,dir] | 15,5 } ]
[ BLKSZ = n ]
[ STORC= storclas ]
[ MGMTC= mgmtclas ]
[ DATAC= dataclas ]
[ VOL= volser ]
[ UNIT= { unit | SYSALLDA} ]
[ RECFM= { F | FB | V | VB } ]
```

DSN=*datasetname*

Specifies the target data set name. If the data set is partitioned, the member name must be included or the data set is corrupted.

You can use the following symbolics in the *datasetname* parameter:

- &USERID—Requesting user ID
- &DAY—Day of week (such as MON)
- &YYYY—Year
- &YY—Year
- &MM—Month
- &MON—Month (such as JAN,FEB)
- &DD—Day
- &HHMMSS—Time
- &HH—Hour
- &MIN—Minute
- &SYSID—System ID
- &SYSNAME—System name
- &JOBNAME—Job name
- &JOBID—Job ID
- &NMID—Region ID
- &NMDID—Region domain ID (DID)
- &GRPNAME—Sysplex name

Symbolics are delimited by a period (.) or another symbolic (that is, &YY&MM. is the same as &YY.&MM.). Symbolics are also allowed in a member name.

For example,

```
DSN=NM.&SYSID..&USERID..D&YY&MM&DD..T&HHMMSS..DATA
```

is converted to

```
DSN=NM.SYSA.MYUSER.D040915.T144505.DATA
```

DISP={ SHR | OLD | NEW | MOD }

Specifies the disposition of the output data set.

- SHR specifies shared use of the data set.
- OLD specifies exclusive use of the data set.
- NEW allocates a new data set.
- MOD appends the output in the file.

Default: SHR.

LRECL={ *n* | 80 }

Specifies the output record length.

Limits: 1 to 250

Default: 80.

SKIP0={ NEWLINE | DISCARD | DESTRUCTIVE | NONDESTRUCTIVE }

Specifies how to process a data line with a skip amount of zero.

- NEWLINE creates a new line of data.
- DISCARD discards the line of data.
- DESTRUCTIVE causes the data to replace the existing data line.
- NONDESTRUCTIVE overlays the data on the existing data line, but only where blanks were present on the existing data line. No existing non-blank characters are modified.

Note: The PSM print options NEWPAGE and USCORE are ignored by the procedures

Default: NEWLINE.

The following additional parameters are applicable when DISP=NEW is specified:

CYL=*pri,sec,dir*

Primary and secondary space allocation values are in cylinders. If a partitioned data set is used, specifies the number of directory blocks.

TRK=*pri,sec,dir*

Primary and secondary space allocation values are in tracks. Number of directory blocks if partitioned data set.

Default: TRK=15,5.

BLKSZ=*blocksize*

Specifies the block size.

STORC=*storclas*

Specifies the storage class.

MGMTC=*mgmtclas*

Specifies the management class.

DATAC=*dataclas*

Specifies the data class.

VOL=*volser*

Specifies the volume serial number.

UNIT= { *unit* | SYSALLDA }

Specifies the unit.

Default: SYSALLDA if volser is specified.

RECFM= { F | *FB* | V | VB }

Record format.

Default: FB.

Example: Printer Exit Definition

This example directs the output for a PSM print request, assigned to the printer named DSEXIT, to the member TEST1 in the data set PROD.PSM.DATA. The record length of this data set is 80. Overlay lines in the data are removed.

Printer Name:	DSEXIT
Type:	EXIT
Description:	Print to a data set
Lower Case:	YES
Line Limit:	0
Form Name:	FORM0
Exit Name:	$PSDS81Z
Exit Data:	DSN=PROD.PSM.DATA(TEST1) LRECL=80 SKIP0=DISCARD

Note: Previous references to parameters WKVOL, CYL, and LIST in the Exit data are no longer required. You must remove them from the printer definition prior to using $PSDS81Z or $PSDS81X, or the print request fails.

Print-to-Email

The $PSEMAIL printer definition lets you email the output of a printing request, which can be either as an attachment or in the body of the email. When the output is sent as an attachment, the email uses the PS8803 message as its body and the PS8804 message as its salutation:

```
Data attached for email_subject

Yours,
user_name
```

user_name

Displays the sender name defined in UAMS.

You can maintain these messages from the Message Definition List panel. The shortcut to the panel is /CASMSG.

Note: For information about how to maintain messages, see the *Managed Object Development Services Programmer and Administrator Guide*.

Appendix A: MAI-FS Sample Subsystem Definitions

These examples show the way in which model 2 LU2 MAI-FS LUs can be defined to CICS and IMS. Remember to define all LU names likely to be used to create a session with these applications as a separate terminal to that application.

Note: For information about definition requirements and for guidance on customizing the definitions to your own requirements, see the appropriate subsystem guides.

Note: Do not include the LOGMODE parameter for either of these subsystems in the definitions.

This section contains the following topics:

CICS

```
DFHTCT TYPE=TERMINAL
       ACCMETH=VTAM
       BRACKET=YES
       BUFFER=buffer-size
       BMSFEAT=(noroute,norouteall)
       GMMSG=YES
       NETNAME=NMMAF001
       OPERID=id
       OPERPRI=code
       PGESIZE=(lines,columns)
       PGESTAT=autopage|page
       RELREQ=(YES,YES)
       RUSIZE=value
       TIOAL=(256,2000)
       TRMIDNT=term
       TRMMODL=2
       TRMSTAT=TRANSCEIVE
       TRMTYPE=LUTYPE2
```

This definition provides support for an MAI-FS session that uses the LU name NMMAF001.

IMS

```
TERMINAL  NAME=NMMTO
          TYPE=3270-A2
          SIZE=(24,80)
          FEATURE=IGNORE
          OPTIONS=(options)
          OUTBUF=2500

NAME      (lterm,MASTER)
```

This definition provides support for an MAI-FS session that uses the LU name NMMTO. If this LU name is used on an MAI-FS session, the functions of the master terminal will be acquired.

```
TERMINAL NAME=NMMAF001
TYPE=3270-A2
SIZE=(24,80)
FEATURE=IGNORE
OPTIONS=(options)
OUTBUF=2500

NAME   lterm
```

This definition provides support for an MAI-FS session that uses the LU name NMMAF001, which could be used for general operations and transaction execution.

Appendix B: MAI-FS Mode Table and Bind Checks

The supplied MAI-FS logmode table, MAIFMODE, is assembled and linked during installation. The table accurately depicts the capabilities of MAI-FS and results in the most efficient use of a session (for instance, by minimizing the number of responses). It is specified on all MAI-FS VTAM APPL statements using the MODETAB=MAIFMODE operand.

The following tables show the checks MAI-FS performs on bind parameters at session initiation. The bits shown are those parameters validated by MAI; bits not shown are not validated. MAI refuses sessions with invalid bind parameters.

This section contains the following topics:

LU Type 0 Sessions

Byte	Bit	Setting	Meaning
2	all	X’02’	FMPROF
3	all	X’02’	TSPROF
4			PRIPROT
	0	B’0’	Single request chains
	2–3	B’00’	Invalid
		B’01’	Exception response
		B’10’	Definite response
		B’11’	Exception or definite response
	6	B’0’	Compression not used

Byte	Bit	Setting	Meaning
5			SECPROT
	0	B’0’T	Single request chains
	1	B’1’	Delayed request mode
	2–3	B’00’	Invalid
		B’01’	Exception response
		B’10’	Definite response
		B’11’	Exception or definite response
	6	B’0’	Compression not used
	7	B’0’	End bracket not sent
6			COMPROT1
	1	B’0’	Function Management Headers not used
	3	B’0’	Bracket termination rule 2
	4	B’0’	Alternate code not used
7			COMPROT 2
	0–1	B’00’	Full duplex
	2	B’0’	Contention loser recovers
	3	B’0’	Primary is contention loser

LU Type 2 Sessions

The secondary logical unit maximum RUSIZE is honored, unless it is greater than 2048 bytes, or less than 64 bytes, in which case a maximum RU size of 2048 bytes is used.

Byte	Bit	Setting	Meaning
2	all	X’03’	FMPROF
3	all	X’03’	TSPROF

Byte	Bit	Setting	Meaning
4			PRIPROT
	0	B’0’	Immediate request mode
	2–3	B’00’	Invalid
		B’01’	Exception response
		B’10’	Definite response
		B’11’	Exception or definite response
	6	B’0’	Compression not used
	7	B’1’	End bracket can be sent
5			SECPROT
	0	B’1’	Multiple request chains
	2–3	B’00’	Invalid
		B’01’	Exception response
		B’10’	Definite response
		B’11’	Exception or definite response
	6	B’0’	Compression not used
6			COMPROT1
	1	B’0’	Function Management Headers not used
	2	B’1’	Bracket are used
	3	B’1’	Bracket termination rule 1
	4	B’0’	Alternate code not used
7			COMPROT 2
	0–1	B’10’	Flip-flop mode
	2	B’0’	Contention loser recovers
	3	B’0’	Primary is contention loser
14	all	X’02’	Logical unit Type 2
20–25	all		Screen sizes specified must be supportable by physical terminal. Unspecified viewport size is acceptable.

Appendix C: MAI-FS Operational Scenario

This scenario gives examples of VTAM and MAI-FS definitions that would be necessary in a site with the following configuration and requirements:

- Two CPUs, linked using MSNF.
- Two SOLVE regions, one named NMP running on the production computer, the other named NMT running on the test computer.
- TSO running on both computers, one named TSOP, the other TSOT.
- IMS running on both computers, one named IMSP, the other IMST.
- A network operator uses a terminal logged on to NMP to control VTAM in both computers, and to operate, using MAI-FS, IMS master terminals to both IMS systems from that SOLVE terminal.
- Authorized personnel can log on to either SOLVE region and create MAI-FS sessions with the TSO or IMS of their choice. A maximum of three MAI-FS sessions from each SOLVE region is allowed.

This section contains the following topics:

Production Computer

This section contains examples for the production computer.

VTAM Definitions

```
MAFP001 APPL MODETAB=MAIFMODE,EAS=1
MAFP002 APPL MODETAB=MAIFMODE,EAS=1
MAFP003 APPL MODETAB=MAIFMODE,EAS=1
MAFMTOP APPL MODETAB=MAIFMODE,EAS=1
MAFMTOT APPL MODETAB=MAIFMODE,EAS=1

MAFT001 CDRSC CDRM=TCDRM
MAFT002 CDRSC CDRM=TCDRM
MAFT003 CDRSC CDRM=TCDRM
```

SOLVE Definitions

```
SYSPARMS MAIFPREF=MAFP ID=NMP
DEFLOGON IN=IMSP
DEFLOGON IN=IMST
DEFLOGON IN=TSOP
DEFLOGON IN=TSOT
```

IMSP Definitions

```
TERMINAL NAME=MAFMTOP,TYPE=3270-A2,SIZE=(24,80) ..etc..
NAME (MAFMTOP,MASTER)
TERMINAL NAME=MAFP001,TYPE=3270-A2,SIZE=(24,80) ..etc..
NAME MAFP001
TERMINAL NAME=MAFP002,TYPE=3270-A2,SIZE=(24,80) ..etc..
NAME MAFP002 T
ERMINAL NAME=MAFP003,TYPE=3270-A2,SIZE=(24,80) ..etc..
NAME MAFP003
TERMINAL NAME=MAFT001,TYPE=3270-A2,SIZE=(24,80) ..etc..
NAME MAFT001
TERMINAL NAME=MAFT002,TYPE=3270-A2,SIZE=(24,80) ..etc..
NAME MAFT002 T
ERMINAL NAME=MAFT003,TYPE=3270-A2,SIZE=(24,80) ..etc..
NAME MAFT003
```

Network Operator

- Creates MAI-FS session with IMSP, specifying NODE NAME = MAFMTOP
- Creates MAI-FS cross-domain MAI-FS session with IMST, specifying NODE NAME = MAFMTOT.

Other User

- Creates MAI-FS session with IMSP. MAI-FS chooses node MAFP001.
- Creates cross-domain MAI-FS session with IMST. MAI-FS chooses node MAFP002.
- Creates MAI-FS session with TSOT. MAI-FS chooses node MAFP003.

Test Computer

This section contains examples for the test computer.

VTAM Definitions

```
MAFT001 APPL MODETAB=MAIFMODE,EAS=1
MAFT002 APPL MODETAB=MAIFMODE,EAS=1
MAFT003 APPL MODETAB=MAIFMODE,EAS=1
MAFP001 CDRSC CDRM=PCDRM
MAFP002 CDRSC CDRM=PCDRM
MAFP003 CDRSC CDRM=PCDRM
MAFMTOT CDRSC CDRM=PCDRM
```

SOLVE Definitions

```
SYSPARMS MAIFPREF=MAFT ID=NMT
DEFLOGON IN=IMSP
DEFLOGON IN=IMST
DEFLOGON IN=TSOP
DEFLOGON IN=TSOT
```

IMST Definitions

```
TERMINAL NAME=MAFMTOT,TYPE=3270-A2,SIZE=(24,80) ..etc..
NAME (MAFMTOT,MASTER)
TERMINAL NAME=MAFP001,TYPE=3270-A2,SIZE=(24,80) ..etc..
NAME MAFP001
TERMINAL NAME=MAFP002,TYPE=3270-A2,SIZE=(24,80) ..etc..
NAME MAFP002
TERMINAL NAME=MAFP003,TYPE=3270-A2,SIZE=(24,80) ..etc..
NAME MAFP003
TERMINAL NAME=MAFT001,TYPE=3270-A2,SIZE=(24,80) ..etc..
NAME MAFT001
TERMINAL NAME=MAFT002,TYPE=3270-A2,SIZE=(24,80) ..etc..
NAME MAFT002
TERMINAL NAME=MAFT003,TYPE=3270-A2,SIZE=(24,80) ..etc..
NAME MAFT003
```

User

- Creates MAI-FS session with IMST.

  MAI-FS chooses node MAFT001.
- Creates cross-domain MAI-FS session with IMSP.

  MAI-FS chooses node MAFT002.
- Creates MAI-FS session with TSOT.

  MAI-FS chooses node MAFT003.

Appendix D: MAI Installation Exit MAIEX02

This section contains the following topics:

MAIEX02

MAI can invoke the MAIEX02 exit at points in exit initialization and termination, and in MAI session initiation and termination. The exit can validate the various parameters used on a session, refuse the session, or change the parameters. Session accounting information is presented to the exit at session end, and can be used to monitor application loading and throughput.

For systems such as IMS where security or operational requirements necessitate connection to a particular IMS LU name, the MAIEX02 exit can force such a connection or suggest a node name prefix to use. This technique can also be useful as a means of limiting the number of terminals that are defined to a subsystem. If all users use MAI, then only the MAI nodes need be defined to the subsystem and only as many as the maximum number of concurrent users need be defined.

The exit must be link edited and placed into a load library accessible to CA SOLVE:Access. The exit must be link edited with the RENT option.

The name of the link edited exit module is identified to CA SOLVE:Access by the SYSPARMS MAIEX02= operand.

The exit is typically written in assembler and can perform any processing required, including SVC calls and WAIT macros. It operates under a subtask that is independent of the main CA SOLVE:Access task.

Note: Although executed under a subtask, the termination call to the exit at system close down is serialized and the termination call must complete before CA SOLVE:Access resumes its termination processing.

Registers on Entry to the Exit

When the exit is invoked, Register 1 contains the address of a fullword, which in turn contains the address of a communications area containing various parameters. This communications area can be mapped using a supplied macro, called $NMMAEX2. This macro provides a DSECT expansion to perform the mapping and detailed information about the content of each field.

Standard linkage conventions apply. On entry, the exit must save the contents of all registers (Register 13 contains the address of a save area) and on exit all registers must be restored to their contents on entry, with the exception of Register 15 which should contain a return code.

Exit Correlators

MAIEX02 has available to it a number of correlator areas, each one a fullword in length. These correlator fields can be used for any purpose. For instance, they can contain counts or storage addresses. The exit can supply or change the content of any correlator supplied during any call. The correlator names and the ways in which they are used are as follows:

System Correlator

One System Correlator exists in a SOLVE region. It is supplied in all calls to MAIEX02. If any call updates it, that updated value will be returned in all subsequent calls.

User Correlator

One User Correlator exists for each SOLVE logon session. That is, when a user first logs on to CA SOLVE:Access, that user is allocated a correlator containing hexadecimal zeroes. If an MAI session start, ACB open or session end call updates the User Correlator, that updated value will be returned in all subsequent session start, ACB OPEN and session end calls

Session Correlator

When a user starts an MAI session, that session is allocated a correlator containing hexadecimal zeroes. If a session start or ACB open call updates the Session Correlator, that updated value will be returned in subsequent calls for that session.

Security Exit Correlator

In addition, session start, ACB open and session end calls are provided with the security exit correlator. This correlator is a user-level correlator that can be provided by the SOLVE security exit.

Security Exit Token Correlator

Session start, ACB open and session end calls are provided with the security exit user token. This is a user-level token that can be provided by the SOLVE security exit.

Contents of the Communications Area

The exit has the following types of call, and the content of the communications area depends on the type of call:

- Exit initialization
- Exit termination
- MAI session start
- ACB open
- MAI session end

Exit Initialization Call

The Exit Initialization call is made when a SYSPARMS MAIEX02 command is entered to invoke the exit. The following parameters are contained in the communications area:

- Function code, F‘0’
- Address of the communications area
- ID of this system
- System Correlator
- A work area to be used for any purpose

Until the exit returns from an Initialization call, no MAI sessions are allowed.

Exit Termination Call

The Exit Termination call is made when your product region is about to terminate. The following parameters are contained in the communications area:

- Function code, F‘4’
- Address of the communications area
- ID of this system
- System Correlator
- A work area to be used for any purpose

MAI Session Start Call

The MAI Session Start call is made when CA SOLVE:Access is about to start a new MAI session. The following parameters are contained in the communications area:

- Function code, F'8'
- Address of the communications area
- ID of this SOLVE region
- Security Exit correlator
- User ID of user starting the session
- Node name of terminal from which session is being started
- Session ID of session being started
- Terminal's screen sizes
- Flags to indicate whether the session is MAI-OC or MAI-FS and whether the terminal supports extended color and highlighting data streams (MAI-FS only)
- The MAI-FS privilege class of the user
- Language code of the user
- System Correlator

Items from this point onwards can be modified by the exit and the modified information will be used when the session is created.

- User Correlator
- Contents of the Communications Area
- User Security Token
- Session Correlator
- Any LU name chosen by the user
- MAI node name prefix as designated by a SYSPARMS MAIOPREF command (for MAI-OC), or a SYSPARMS MAIFPREF command or a DEFLOGON command (for MAI-FS)
- Whether the session will use the Fast Jump and Compression options (MAI-FS only)
- Whether ACB sharing will be attempted for this session (MAI-FS only)
- The application with which the session will be started
- The LOGMODE name chosen by the user or MAI
- The length of any user data to be passed to the application at LOGON
- The user data itself
- An area into which the exit may place an error message if the session is to be refused
- A work area to be used for any purpose
- Settings of all the jump keys associated with the session (MAI-FS only)
- An indicator word, into which the exit places a value of F'4' if it requires an ACB open call
- The name of an NCL procedure to perform session script functions, and optional parameters to be passed to it

ACB Open Call

The ACB Open call is made after MAI has successfully opened or allocated the ACB to use for the session. However, the call is only made if the exit returned the appropriate value in the indicator word after the Session Start call. The following parameters are contained in the communications area:

- Function code, F‘16’
- Address of the communications area
- ID of this system
- Security Exit correlator
- User ID of user starting the session
- Node name of terminal from which session is being started
- Session ID of session being started
- Flags to indicate whether the session is MAI-OC or MAI-FS
- The MAI-FS privilege class of the user
- System Correlator
- User Correlator
- User Security Token
- Session Correlator
- Name of the ACB which will be used on this session
- Name of the application with which the session will be started
- A work area to be used for any purpose
- The length of any user data to be passed to the application at LOGON
- The user data

MAI Session End Call

The MAI Session End call is made when an MAI session has ended. The following parameters are contained in the communications area:

- Function code, F'12'
- Address of the communications area
- ID of this SOLVE region
- Security Exit correlator
- User ID of user on whose behalf session was started
- Node name of terminal from which session was started
- Application with which session was started
- Session ID of session ending
- Flags to indicate whether session is MAI-OC or MAI-FS and whether any WAIT=PERM specification for the session is to be canceled
- System Correlator
- User Correlator
- User Security Token
- Session Correlator
- A work area to be used for any purpose
- The count of inbound bytes over the duration of the session
- The count of outbound bytes over the duration of the session
- The count of inbound RUs over the duration of the session
- The count of outbound RUs over the duration of the session

Return Codes from the Exit

When the exit returns to MAI-FS, Register 15 should contain a return code for the Exit Initialization call and the Session Start call.

Exit Initialization

Register 15 should contain zero if exit initialization was successful. A non-zero value in register 15 indicates that initialization was not successful. In this case, no MAI sessions will be allowed. Any attempts to start an MAI session will be rejected with an appropriate error message.

Session Start

Register 15 should contain zero if the session should proceed using the parameters in the communications area. Any value other than zero indicates that the session is to be refused and the message placed in the message field of the communications area displayed at the user's terminal. If this field is blank, the following default message will be displayed:

```
SESSION REFUSED BY INSTALLATION EXIT
```

How the Exit Is Called

MAIEX02 can be started using the SYSPARMS MAIEX02= command, naming the program that is to act as the exit. When this command is executed as part of the NMINIT procedure, usually during CA SOLVE:Access initialization, an Exit Initialization call is made. Subsequent MAI session requests then call the exit. If the exit is not named during initialization, but is invoked using command from a SOLVE terminal, any existing users with MAI sessions may have to log off CA SOLVE:Access before any calls are made to the exit for that user.

During the testing phases of the exit, you may want to force a new Initialization Call or to invoke an entirely different program. You can accomplish these actions entering another SYSPARMS MAIEX02= command, specifying the same or a different program name. When the command is entered, an Initialization Call is made, with a zero system correlator and any existing user correlators are zeroed. MAI also ensures that if any MAI sessions are currently in progress, the ending of those sessions does not cause a Session End call to the old or the new exit.

Reentrant Code

You must write the exit to conform to the following rules:

- It is written in reentrant code
- It does not modify any storage location within itself
- Any working storage required other than that provided in the communications area is obtained using GETMAIN.

Failure to observe these conditions will result in an abend in the processing task and the session request will fail. Other processing is unaffected.

Storage Subpools

It is possible for the exit to maintain information across calls in GETMAIN storage, remembering the addresses of the storage in the various correlators. If this technique is to be utilized, the storage must be obtained in subpool 50. Storage obtained in any other subpool may be automatically freed when the exit returns to MAI or, in the case of storage obtained on a Session Start call, when that MAI session ends.

It is the responsibility of the exit to free any GETMAIN storage when necessary. MAI will never free storage obtained by the exit.

Exit Serialization

Because MAIEX02 is run in a subtask of CA SOLVE:Access, it is possible for it to run concurrently for two or more different users. That is, while it is processing, say, a Session Start call for one user, it could be called to process a Session End call for another. This could lead to complications where, for instance, the exit is maintaining a control block structure or changing a correlator because the subsequent call may 'interrupt' processing in the first call.

MAI provides two levels of serialization to overcome these problems, governed by the SYSPARMS MAIEX02S= command. MAIEX02S=SYSTEM, the default, ensures that calls to the exit are totally serialized so that, in the above example, the Session End call would not be made until the Session Start call had completed. This level is used where, for instance, the exit is maintaining a control block structure or where the various calls update the System Correlator. It ensures that a System Correlator set by one call is always passed to the next call.

The only drawback with this level of serialization is that the starting and ending of MAI sessions is single-threaded so that only one can proceed at a time. However, as the path through MAIEX02 will almost always be short and fast, this should not create a problem.

The second level of serialization, MAIEX02S=USER, ensures that calls to the exit are serialized at a user level only. That is, the exit can concurrently process multiple calls for different users, but calls for any one user will be serialized. This level protects the User Correlator, ensuring that if set by one call it is presented to the next call for that user correctly. However, it does not protect the System Correlator or any control block structure the exit may maintain.

MAIEX02S=USER provides processing overlap advantages over the MAIEX02S=SYSTEM option and can be used with safety if the exit is purely validating or changing session parameters.

Sample Exit

A sample exit, MAIEX02, is distributed in the *dsnpref*.SM40MS.MSSAMP library. It is recommended that this sample be assembled and studied prior to attempting to write an exit. The macro $NMMAEX2 must be available for this assembly to function correctly.

Counting of MAI Sessions

The sample shows a way in which the correlators can be used to count the number of MAI sessions a user has and the total number at any one time. It then limits the number of sessions allowed and rejects sessions if the counts pass a predetermined level. Because Session Start and End calls to the sample exit update the System Correlator, it requires the MAIEX02S=SYSTEM serialization level.

Appendix E: MAI Installation Exit MAIEX03

MAI-FS can invoke the installation-written exit MAIEX03 whenever a jump between sessions, or a jump to CA SOLVE:Access is performed. The exit can be used to send an installation defined data stream to the terminal. The exit runs synchronously with respect to the SOLVE main task and therefore should not lose control in any way. Keep processing within the exit to a minimum.

The exit must be link edited and placed into a load library accessible to CA SOLVE:Access.

The name of the link edited exit module is identified to CA SOLVE:Access by the SYSPARMS MAIEX03= operand.

This section contains the following topics:

Registers on Entry to the Exit

When the exit is invoked, register 1 contains the address of a fullword which in turns contains the address of a communications area containing various parameters. This communications area can be mapped using a macro supplied with CA SOLVE:Access called $NMMAEX3. This macro provides a DSECT expansion to perform the mapping and detailed information on the content of each field.

Standard linkage conventions apply. On entry, the exit must save the contents of all registers (register 13 will contain the address of a save area).

Exit Correlators

MAIEX03 has a number of correlator areas, each one a fullword in length, that have been set up by a previous call to the MAIEX02 exit. The MAIEX03 exit cannot change the content of any correlator supplied during any MAIEX03 call. The correlator names and the ways in which they are used are as follows:

System Correlator

One System Correlator exists in a system. This correlator is supplied in all calls to MAIEX03.

Session Correlator

When a user starts an MAI session, that session is allocated a correlator containing hexadecimal zeros. A previous call to MAIEX02 can modify this correlator.

Security Exit Correlator

Session start, ACB open, and session end calls are provided with the security exit correlator. The SOLVE security exit provides this user level correlator.

Note: For more information, see the *Security Guide.*

Contents of the Communication Area

The passed communications area contains the following information:

- A flag to indicate the type of jump in progress where a value of:
  - 0 = a jump away from a session
  - 4 = a jump to a session
  - 8 = a jump to a SOLVE application
- Address of the communications area.
- ID of this system.
- A security exit correlator.
- The user ID of this user.
- The node name of the terminal.
- The name of the application being jumped away from or jumped to. This field will be set to blanks if this is a jump to SOLVE call.
- The MAI ACB name used by this session. This field will be set to blanks if this is a jump to SOLVE call.
- System Correlator.
- Session Correlator. This field will contain zeros if this is a jump to SOLVE call.
- Exit work area.

Registers on Return From the Exit

The contents of the registers on return from the exit will depend on the function to be performed by MAI as described below:

- If register 15 is set to 0, the session jump will proceed and no action will be performed on behalf of the exit. All other registers should be returned to the value contained on entry to the exit.
- If register 15 is set to 4, the session jump will proceed after sending an installation defined data stream to the terminal. In this case, registers 1 and 0 should contain the address and length of the data stream respectively. All other registers should be returned to the value contained on entry to the exit.

Serial Reuseability

Because the exit is run from the SOLVE main task and should never lose control, it is not absolutely necessary for the exit code to be re-entrant. It is, however, necessary for the exit code to be serially reuseable. That is, to restore any storage locations within the exit that were modified during the call to the values contained at the time of the call. This is necessary to stop independent calls to the exit from having any effect on one another. A work area is supplied as part of the communication area and, where possible, should be used for any exit storage requirements.

Sample Exit

A sample exit, MAIEX03, is installed with your product and copied to *dsnpref*.SM40MS.MSSAMP. It is recommended that this sample be assembled and studied prior to attempting to write an exit. The macro $NMMAEX3 must be available for this assembly to function correctly.

Appendix F: MAI Session List NCL Interface

This section contains the following topics:

About the MAI Session List NCL Interface

MAI Session Lists define the applications that users of CA SOLVE:Access are allowed to access and the parameters for the session setup. MAI session lists are stored on the ACDB UDB. Each MAI session list comprises a number of session entries that are uniquely defined by the session identifier.

The MAI Session List interface retrieves, modifies, and stores MAI session lists and their entries. It provides a logical interface that hides the physical file format. By using this interface, code is protected from file changes that future versions of CA SOLVE:Access may require.

$MASD00F is an NCL procedure that provides a callable interface for MAI session list maintenance. The API accepts keyword parameters to control operation and shares variables and MDOs to pass session list and file information.

The general format of the procedure is as follows:

```
&CALL PROC=$MASD00F SHARE=($MASD>) + PARMS=(OPT=option,+ keyword=value,…)
```

The following options are available:

- OPT=INIT - establishes file access
- OPT=TERM - terminates file access
- OPT=USERGET - retrieves a session list
- OPT=USERADD - creates a new session list
- OPT=USERCOPY - copies all or selected sessions to another session list
- OPT=USERPUT - adds or replaces a session list
- OPT=USERDEL - deletes a session list
- OPT=USERINIT - sets up to build a new session list
- OPT=USERLOCK - synchronizes access to session lists
- OPT=DEFGET - retrieves session entry details
- OPT=DEFADD - creates a new session entry
- OPT=DEFPUT - adds or replaces a session entry
- OPT=DEFDEL - deletes a session entry
- OPT=DEFINIT - initializes a new session entry with the system defaults

The $MASD00F API provides the following levels of access based on user authority:

Authority Level 4 or higher

System Support Access = Y

Access: Read, Update, Delete

Authority Level3

System Support Access = Y

Access: Read, Update

Authority Level2

System Support Access = Y

Access: Read

Authority Level 1,0

Access: Update own session list

Authority Level -

System Support Access = N

Access: Update own session list

$MASD00F OPT=INIT

This call establishes a connection with the UDB containing the MAI session lists.

This call has the following format:

```
&CALL PROC=$MASD00F SHARE=($MASD>) +
      PARMS=(OPT=INIT,+
    [ FILEID={ ACDB | file-id }])
```

On return, &RETCODE indicates the following:

0

Function successful.

8

&FILE OPEN failed. &SYSMSG is set.

$MASD00F OPT=TERM

This call terminates a connection with the UDB that contains the MAI session lists.

This call has the following format:

```
&CALL PROC=$MASD00F SHARE=($MASD>) +
      PARMS=(OPT=TERM,+
    [ FILEID={ ACDB | file-id }])
```

On return, &RETCODE indicates the following:

0

Function successful.

8

&FILE CLOSE failed. &SYSMSG is set.

$MASD00F OPT=USERINIT

This call initializes an MAI session list. Subsequent OPT=DEFADD,FILEPUT=NO calls can be used to build the sessions in the list. The session list can then be written to the file using FILEPUT=YES on the last OPT=DEFADD or by an OPT=USERPUT/USERADD call.

This call has the following format:

```
&CALL PROC=$MASD00F SHARE=($MASD>) +
      PARMS=(OPT=USERINIT,+
      USERKEY=slistname)
```

The following keyword controls the operation:

USERKEY

Specifies the session list name.

On return, &RETCODE indicates the following:

0

Function successful.

&$MASD#USER contains the key of the MAI session list. MDO $MASD#MDO mapped by $ACMPSDL contains the empty session list.

$MASD00F OPT=USERGET

This call retrieves an MAI session list. Subsequent OPT=DEF*xxxx* calls can then be used to access or modify the session list.

This call has the following format:

```
&CALL PROC=$MASD00F SHARE=($MASD>) +
      PARMS=(OPT=USERGET,+
    [ FILEID={ ACDB | file-id },+ ]
    [ USERKEY=slistname,+ ]
    [ GENLEN=keylength,+ ]
    [ FILEOPT={ KEQ | KGE | KLE | KGT | KLT },+ ]
    [ LOCKREC={ NO | YES }])
```

The following keywords control the read operation:

FILEID

Optionally specifies the UDB name of the file to access. The default is the file used on the OPT=INIT call.

Note: This operand is ignored if USERKEY is not specified.

USERKEY

Specifies a full or partial key for the session list name.

If a generic read operation is used, specify USERKEY on the first read to establish the file position and do not specify a key on subsequent read operations.

GENLEN

Specifies the generic key length. A value of 0 indicates all records are read.

FILEOPT

Specifies the type of read operation as for the NCL &FILE GET verb.

LOCKREC

YES indicates an NCL lock is required to synchronize access to this record. The lock has a primary name of $MA-&$MASD#FILEID and a minor name of the session list name. Use this option if you want to update or delete the record. Return code 8 is set if the lock cannot be obtained and &SYSMSG contains an MA3031 message. If the current procedure already holds the lock, return code 16 is set.

On return, &RETCODE indicates the following:

0

Function successful.

&$MASD#USER contains the key of the MAI session list. MDO $MASD#MDO mapped by $ACMPSDL contains the session list.

4

Record not found. If the operation is a direct read, &SYSMSG is set.

8

An error occurred. &SYSMSG is set.

16

Lock error. &SYSMSG is set.

$MASD00F OPT=USERADD

This call adds an MAI session list. If COPYKEY is not specified, the session list is passed in the MDO $MASD#MDO mapped by $ACMPSDL.

This call has the following format:

```
&CALL PROC=$MASD00F SHARE=($MASD>) +
      PARMS=(OPT=USERADD,+
      USERKEY=slistname,+
    [ COPYKEY=copyname,+ ]
    [ FILEID={ ACDB | file-id },+ ]
    [ LOCKFREE={ NO | YES },+ ]
    [ LOCKREC={ NO | YES }])
```

The following keywords control the read operation:

FILEID

Optionally specifies the UDB name of the file to access. The default is the file used on the OPT=INIT call.

USERKEY

Specifies the session list name.

COPYKEY

Specifies an existing session list record to insert for the specified user.

LOCKFREE

YES specifies that a previously obtained NCL lock for this session list is freed after the file update.

LOCKREC

YES indicates an NCL lock is required to synchronize access to this record. The lock has a primary name of $MA-&$MASD#FILEID and a minor name of the session list name. Return code 8 is set if the lock cannot be obtained and &SYSMSG contains an MA3031 message. If the current procedure already holds the lock, return code 16 is set.

On return, &RETCODE indicates the following:

0

Function successful.

4

Record exists or COPYKEY session list not found. &SYSMSG is set.

8

Unable to obtain lock. &SYSMSG is set.

16

Lock error. &SYSMSG is set.

20

&FILE PUT error. &SYSMSG is set and also written to the activity log.

$MASD00F OPT=USERCOPY

This call updates an existing MAI session list. If COPYKEY is not specified, the session list is passed in the MDO $MASD#MDO mapped by $ACMPSDL.

This call has the following format:

```
&CALL PROC=$MASD00F SHARE=($MASD>) +
      PARMS=(OPT=USERCOPY,+
      USERKEY=slistname,+
    [ COPYKEY=copyname,+ ]
    [ COPYOPT=( RPLSESSLIST | RPLMATCHSESS ),+ ]
    [ FILEID={ ACDB | file-id },+ ]
    [ LOCKFREE={ NO | YES },+ ]
    [ LOCKREC={ NO | YES }, + ]
    [ SESSIONS={ ALL | LIST }])
```

The following keywords control the read operation:

FILEID

Optionally specifies the UDB name of the file to access. The default is the file used on the OPT=INIT call.

USERKEY

Specifies the session list name to update.

COPYKEY

Specifies an existing session list record to use as the data source.

COPYOPT

Specifies the type of copy operation. If no option is specified, the input sessions are used to update the target session list. RPLSESSLIST specifies that the input list replaces the target session list. RPLMATCHSESS updates existing sessions in the target session list where the session IDs match.

Note: The SESSIONS option can be used to filter the input session list.

LOCKFREE

YES specifies that a previously obtained NCL lock for this session list is freed after the file update.

LOCKREC

YES indicates an NCL lock is required to synchronize access to this record. The lock has a primary name of $MA-&$MASD#FILEID and a minor name of the session list name. Return code 8 is set if the lock cannot be obtained and &SYSMSG contains an MA3031 message. If the current procedure already holds the lock, return code 16 is set.

SESSIONS

Specifies which sessions are eligible to copy into the target session list. ALL makes available all sessions in the COPYKEY session list. LIST specifies that variables contain the session IDs to make available for copying. &$MASD$RCNT is the number of sessions to copy and the set of &$MASD$RID*n* variables contain the session IDs to copy.

On return, &RETCODE indicates the following:

0

Function successful.

4

Record not found or COPYKEY session list not found. &SYSMSG is set.

8

Unable to obtain lock. &SYSMSG is set.

16

Lock error. &SYSMSG is set.

20

&FILE PUT error. &SYSMSG is set and also written to the activity log.

$MASD00F OPT=USERPUT

This call adds or replaces an MAI session list. The session list is passed in the MDO $MASD#MDO mapped by $ACMPSDL.

This call has the following format:

```
&CALL PROC=$MASD00F SHARE=($MASD>) +
      PARMS=(OPT=USERPUT,+
      USERKEY=slistname,+
    [ FILEID={ ACDB | file-id },+ ]
    [ LOCKFREE={ NO | YES },+ ]
    [ LOCKREC={ NO | YES }])
```

The following keywords control the read operation:

FILEID

Optionally specifies the UDB name of the file to access.

Default: File used on the OPT=INIT call

USERKEY

Specifies the session list name.

LOCKFREE

YES specifies that a previously obtained NCL lock for this session list is freed after the file update.

LOCKREC

YES indicates an NCL lock is required to synchronize access to this record. The lock has a primary name of $MA-&$MASD#FILEID and a minor name of the session list name. Return code 8 is set if the lock cannot be obtained and &SYSMSG contains an MA3031 message. If the current procedure already holds the lock, return code 16 is set.

On return, &RETCODE indicates the following:

0

Function successful.

8

Unable to obtain lock. &SYSMSG is set.

16

Lock error. &SYSMSG is set.

20

&FILE PUT error. &SYSMSG is set and also written to the activity log.

$MASD00F OPT=USERLOCK

This call obtains or frees an NCL lock on an MAI session list.

This call has the following format:

```
&CALL PROC=$MASD00F SHARE=($MASD>) +
      PARMS=(OPT=USERLOCK,+
      USERKEY=slistname,+
    [ FILEID={ ACDB | file-id,+ ]
    { LOCKFREE={ NO | YES } |
      LOCKREC={ NO | YES }})
```

The following keywords control the lock operation:

FILEID

Optionally specifies the UDB name of the file to access.

Default: File used on the OPT=INIT call

USERKEY

Specifies the session list name.

LOCKFREE

YES specifies that a previously obtained NCL lock for this session list is freed.

LOCKREC

YES indicates an NCL lock is required to synchronize access to this record. The lock has a primary name of $MA-&$MASD#FILEID and a minor name of the session list name. Return code 8 is set if the lock cannot be obtained and &SYSMSG contains an MA3031 message. If the current procedure already holds the lock, return code 16 is set.

On return, &RETCODE indicates the following:

0

Function successful.

8

Unable to obtain lock. &SYSMSG is set.

16

Lock error. &SYSMSG is set.

$MASD00F OPT=USERDEL

This call deletes an MAI session list.

This call has the following format:

```
&CALL PROC=$MASD00F SHARE=($MASD>) +
      PARMS=(OPT=USERDEL,+
    [ FILEID={ ACDB | file-id },+ ]
      USERKEY=slistname,+
    [ LOCKFREE={ NO | YES }])
```

The following keywords control the read operation:

FILEID

Optionally specifies the UDB name of the file to access.

Default: File used on the OPT=INIT call

USERKEY

Specifies the full key for the session list name.

LOCKFREE

YES indicates the NCL lock used to synchronize access to this record is released.

On return, &RETCODE indicates the following:

0

Function successful.

4

Record not found.

20

An error occurred. &SYSMSG is set.

$MASD00F OPT=DEFGET

This call accesses a session entry in the current session list. The session list is passed in the $MASD#MDO MDO. On return, the set of &$MASD> session entry variables is updated.

This call has the following format:

```
&CALL PROC=$MASD00F SHARE=($MASD>) +
      PARMS=(OPT=DEFGET,+
    [ SESSKEY=sessid,+ ]
    [ FILEOPT={ TOP | END | FWD | BWD | KEQ ]})
```

The following keywords control the get operation:

SESSKEY

Specifies the session identifier of the entry to return when FILEOPT=KEQ is also specified.

FILEOPT

Controls the type of access. Specify TOP to retrieve the first session entry, END for the last. Use BWD and FWD to move to the next entry. Use KEQ in combination with SESSKEY to retrieve a specific session entry by its session identifier.

On return, &RETCODE indicates the following:

0

Function successful. Session entry variables are set.

4

Session entry not found or end of list reached.

The following session entry variables are set when a session entry is retrieved:

$MASDSESSID

Session identifier, 1 through 8 characters

$MASDLOGON

Logon data, 1 through 50 characters

$MASDLUNAME

Specific virtual terminal name, 1 through 8 characters

$MASDFWDK1

Jump forward key 1: F1 to F24, PA1, PA2, PA3, CLEAR, ATTN

$MASDFWDK2

Jump forward key 2: F1 to F24, PA1, PA2, PA3, CLEAR, ATTN

$MASDREVK1

Jump reverse key 1: F1 to F24, PA1, PA2, PA3, CLEAR, ATTN

$MASDREVK2

Jump reverse key 2: F1 to F24, PA1, PA2, PA3, CLEAR, ATTN

$MASDMENUK1

Jump to menu key 1: F1 to F24, PA1, PA2, PA3, CLEAR, ATTN

$MASDMENUK2

Jump to menu key 2: F1 to F24, PA1, PA2, PA3, CLEAR, ATTN

$MASDSWAPK1

Swap windows key 1: F1 to F24, PA1, PA2, PA3, CLEAR, ATTN

$MASDSWAPK2

Swap windows key 2: F1 to F24, PA1, PA2, PA3, CLEAR, ATTN

$MASDSISPK

SIS print screen key: F1 to F24, PA1, PA2, PA3, CLEAR, ATTN

$MASDSISMK

SIS menu key: F1 to F24, PA1, PA2, PA3, CLEAR, ATTN

$MASDWAIT

Wait option, YES or NO

$MASDREST

Restart option, YES or NO

$MASDFASTJMP

Fast jump option, YES or NO

$MASDCOMPRSS

Compression option, YES or NO

$MASDLOGMODE

Override logmode name, 1 through 8 characters

$MASDSCRIPT

Script name and parameters, 1 through 26 characters

$MASD00F OPT=DEFADD

This call adds a session entry to the current session list. The session list is passed in the $MASD#MDO MDO. The new session details are passed in the set of &$MASD> session entry variables as described for OPT=DEFGET.

Note: The session entry values are not validated. Invalid key settings are ignored, and incorrect session details can result in an unusable session definition.

This call has the following format:

```
&CALL PROC=$MASD00F SHARE=($MASD>) +
      PARMS=(OPT=DEFADD,+
      SESSKEY=sessid,+
    [ FILEPUT={ YES | NO },+ ]
    [ MOVEOPT={ TOP | END | AFTER },+ ]
    [ MOVEKEY=sessid2 ])
```

The following keywords control the add operation:

SESSKEY

Specifies the session ID of the new session. The ID must be unique in the existing session list.

FILEPUT

Controls whether the updated session list is written back to the file.

MOVEOPT

Specifies where in the session list the new entry is inserted. Use MOVEOPT=AFTER and an existing session ID on the MOVEKEY operand to insert into the session list.

MOVEKEY

Specifies an existing session identifier after which the new session is inserted.

On return, &RETCODE indicates the following:

0

Function successful.

The session entry has been inserted and the session list written back to the file.

4

Error occurred, and &SYSMSG is set.

- Session entry for *sessid* exists.
- *sessid2* is not found in session list.

16

Internal logic error.

Invalid insertion entry number.

$MASD00F OPT=DEFPUT

This call updates an existing session entry in the current session list. The session list is passed in the $MASD#MDO MDO. The new session details are passed in the set of &$MASD> session entry variables as described for OPT=DEFGET.

Note: The session entry values are not validated. Invalid key settings are ignored, and incorrect session details can result in an unusable session definition.

This call has the following format:

```
&CALL PROC=$MASD00F SHARE=($MASD>) +
      PARMS=(OPT=DEFPUT,+
      SESSKEY=sessid,+
    [ FILEPUT={ YES | NO },+ ]
    [ NEWKEY=sessid2 ])
```

The following keywords control the add operation:

SESSKEY

Specifies the session ID of the session. The ID must currently exist in the session list.

NEWKEY

Optionally specifies a new session identifier for the session entry.

FILEPUT

Controls whether the updated session list is written back to the file.

On return, &RETCODE indicates the following:

0

Function successful. The session entry is updated and the session list written back to the file.

4

Error occurred and &SYSMSG is set.

Session entry for *sessid* does not exist

$MASD00F OPT=DEFDEL

This call deletes a session entry in the current session list. The session list is passed in the $MASD#MDO MDO.

This call has the following format:

```
&CALL PROC=$MASD00F SHARE=($MASD>) +
      PARMS=(OPT=DEFDEL,+
      SESSKEY=sessid,+
    [ FILEPUT={ YES | NO }])
```

The following keywords control the get operation:

SESSKEY

Specifies the session identifier of the session entry to delete.

FILEPUT

Controls whether the updated session list is written back to the file.

On return, &RETCODE indicates the following:

0

Function successful. The session entry has been deleted and the session list written back to the file.

4

Session entry not found.

$MASD00F OPT=DEFINIT

This call sets the &$MASD> session entry variables to the default values. The default values are defined in the Default Session Details panel (/ACADMIN.D).

This call has the following format:

```
&CALL PROC=$MASD00F SHARE=($MASD>) +
      PARMS=(OPT=DEFINIT)
```

$MASD00F OPT=DEFMOVE

This call moves a session entry to the current session list. The session list is passed in the $MASD#MDO MDO.

This call has the following format:

```
&CALL PROC=$MASD00F SHARE=($MASD>) +
      PARMS=(OPT=DEFMOVE,+
      BASEKEY=sessid,+
    [ MOVEOPT={ TOP | END | AFTER | BEFORE },+ ]
    [ MOVEKEY=sessid2 ])
```

The following keywords control the add operation:

SESSKEY

Specifies the session ID of the session entry to move.

MOVEOPT

Specifies where in the session list the entry is moved. Use MOVEOPT=AFTER or BEFORE to position relative to an existing session as specified by the MOVEKEY operand.

Default: Move the session to the end of the list.

MOVEKEY

Specifies an existing session identifier for positioning the session entry being moved.

On return, &RETCODE indicates the following:

0

Function successful. The session entry is moved and the session list written back to the file.

4

Error occurred, and &SYSMSG is set.

- Session entry for *sessid* exists.
- *sessid2* is not found in session list.

16

Internal logic error.

Invalid insertion entry number.

$MASD00F API Examples

The following samples are provided as examples of $MASD00F coding:

ACREAD

Generic read using a partial key.

ACCOMP

Generic read of two files and compare existence of session lists.

ACCOPY

Generic read from one file and write to another.

ACDEFADD

Adds a session entry to an existing session list or creates a session list.

Appendix G: Health Checks

This section contains the following topics:

CA Health Checker

The CA Health Checker provides a simple and consistent method for CA products to create health checks to run under the IBM Health Checker for z/OS. The IBM Health Checker for z/OS helps you identify potential problems in your z/OS environment by checking system or product parameters and system status against recommended settings. CA SOLVE:Access health checks are automatically activated on the target system when the product is started on a system with IBM Health Checker for z/OS installed and configured.

The CHECK_OWNER for all CA SOLVE:Access health checks is CA_NM.

Use either CA SYSVIEW or SDSF Health Checker displays to list and view the checks. View messages generated by CA health checks in the MVS System Log.

NM_AC_ACDB

Description

Checks that the region's ACDB database is available. This check runs every 5 minutes.

The ACDB is a critical database for CA SOLVE:Access. When the database is not available, users who log on are not able to access applications.

Best Practice

None.

Parameters accepted

None.

Debug Support

No.

Verbose Support

No.

Reference

None.

Non-exception Messages

The following messages can appear in health checker:

- ACDB file, DSN=*name*, available with options *optionstring*. Session menus are accessible.
- The region is initializing. Check is not relevant at this time.
- The region is shutting down. Check is not relevant at this time.

Exception Messages

If an exception occurs, the following messages are issued as WTOs and written to the SYSLOG:

- CAH0001E The check timed out while waiting for a response to a command.
- NMHAC04E ACDB file, DSN=*name*, not available for CA SOLVE:Access. New users will not be able to access applications.

NM_AC_MAIPOOLS

Description

Checks that sufficient virtual terminal ACBs are available. This check runs every 10 minutes.

The ACDB is a critical database for CA SOLVE:Access. When the database is not available, users who log on are not able to access applications.

Best Practice

Use the EXTAPPLPOOLS parameter group to allow sufficient virtual terminal ACBs.

Parameters accepted

THRESHOLD=*percent_used*

Specifies the ACB usage threshold as a percentage of the total available ACBs. An exception occurs when the threshold is exceeded.

Default: 95

Debug Support

No.

Verbose Support

No.

Reference

See the online help for the EXTAPPLPOOLS parameter group.

Non-exception Messages

The following messages can appear in health checker:

- *n* MAI ACB pools available, *nn*% utilized. *nnn* ACBs are available for use as virtual terminals.
- The region is initializing. Check is not relevant at this time.
- The region is shutting down. Check is not relevant at this time.

Exception Messages

If an exception occurs, the following messages are issued as WTOs and written to the SYSLOG:

- CAH0001E The check timed out while waiting for a response to a command.
- NMHAC06E *n* MAI ACB Pools are *nn*% utilized exceeding the threshold of *mm*%. Access to applications may fail when 100% allocation is reached. *nnn* ACBs are available for use as virtual terminals.

NM_AC_MAITASKS

Description

Checks that sufficient subtasks are available for the workload. This check runs every 10 minutes.

Best Practice

Set the priority of the CA SOLVE:Access address space in the Workload Manager such that the setting is lower than VTAM but higher than the applications being accessed.

Use the MAIPARMS parameter group to increase the number of subtasks.

Parameters accepted

None.

Debug Support

No.

Verbose Support

No.

Reference

See the online help for the MAIPARMS parameter group.

Non-exception Messages

The following messages can appear in health checker:

- *n* session subtasks active. No CA SOLVE:Access subtask performance issues were detected. MAI Session throughput is normal.
- The region is initializing. Check is not relevant at this time.
- The region is shutting down. Check is not relevant at this time.

Exception Messages

If an exception occurs, the following messages are issued as WTOs and written to the SYSLOG:

- CAH0001E The check timed out while waiting for a response to a command.
- NMHAC02E *n* session subtasks active. Excessive queueing detected for *m* CA SOLVE:Access subtasks. MAI session throughput may be impacted.

NM_ACB

Description

Checks that the region's primary ACB is open. This check runs every 5 minutes.

VTAM is required to access the 3270 interface.

Best Practice

None.

Parameters accepted

None.

Debug Support

No.

Verbose Support

No.

Reference

None.

Non-exception Messages

The following messages can appear in health checker:

- This region's primary ACB, *acbname*, is open.
- The region is shutting down. Check is not relevant at this time.

Exception Messages

If an exception occurs, the following messages are issued as WTOs and written to the SYSLOG:

- CAH0001E The check timed out while waiting for a response to a command.
- NMH0106E This region's primary ACB, *acbname*, is not open.

NM_INITIALIZATION

Description

Checks region initialization. The check runs once at region startup. If an exception occurs, the check repeats every 5 minutes until initialization is successful.

Best Practice

Follow the Install Utility procedures in the *Installation Guide* to set up your region, and ensure that the parameters are specified correctly.

Parameters Accepted

None.

Debug Support

No.

Verbose Support

No.

Reference

See the online help for region parameter groups.

Non-exception Messages

The following messages can appear in health checker:

- The region has initialized successfully.
- The region is initializing. Check is not relevant at this time.
- The region is shutting down. Check is not relevant at this time

Exception Messages

If an exception occurs, the following messages are issued as WTOs and written to the SYSLOG:

- CAH0001E The check timed out while waiting for a response to a command.
- NMH0104E Initialization errors have occurred in region *regionname*.

NM_SSI

Description

Checks that the SOLVE SSI SSID is defined and connected. The check runs every 15 minutes.

Best Practice

Ensure that the following conditions are met:

- The SOLVE SSI started task is active.
- The region's SOLVE SSI SSID the value matches the SSID= parameter for the SOLVE SSI started task.

Parameters Accepted

None.

Debug Support

No.

Verbose Support

No.

Reference

None.

Non-exception Messages

The following messages can appear in health checker:

- SOLVE SSI SSID correctly defined and connected. SSID is *ssidname*.
- The region is initializing. Check is not relevant at this time.
- The region is shutting down. Check is not relevant at this time.

Exception Messages

If an exception occurs, the following messages are issued as WTOs and written to the SYSLOG:

- CAH0001E The check timed out while waiting for a response to a command.
- NMH0108E SSID error, no SSID specified.
- NMH0108E SSID error, *ssidname* is not connected.
- NMH0108E SSID error, SSID matches AOM SSID(*ssidname*).

Index

D

E

F

G

H

I

J

L

M

N

O

P

R

S

T

U

V

W